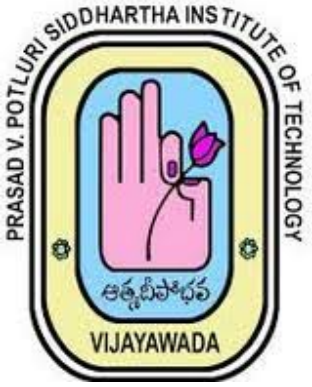
Prasad V. Potluri Siddhartha Institute of Technology

Autonomous, Kanuru,Vijayawada-07

Department of Computer Science and Engineering

Mobile App Development – 20SA8651

Laboratory Manual



AICTE Approved, NBA and NACC A+ accredited, An ISO 9001: 2015 certified Institution, Permanent affiliation to JNTUK, Kakinada.

Institute Vision

To provide rich ambience for Academic and Professional Excellence, Research, Employability skills, Entrepreneurship and Social responsibility.

Institute Mission

To empower the students with Technical knowledge, Awareness of up-to-date technical trends, Inclination for research in the areas of human needs, Capacity building for Employment / Entrepreneurship, Application of technology for societal needs.

Department Vision

To be a center of excellence in academics and research in Computer Science and Engineering and take up challenges for the benefit of society.

Department Mission

Impart professional education through best curriculum in harmony with the industry needs.

Inculcate ethics, research capabilities and team work in the young minds so as to put efforts to the advancement of the nation.

Strive for student achievement and success with leadership qualities and preparing them for continuous learning in the global environment.

Program Educational Objectives

PEO-I:

The graduates of the program will excel in the concepts of basic engineering and advanced concepts of computer science engineering.

PEO-II:

The graduates of the program will be professional in computing industry or pursuing higher studies.

PEO-III:

The graduates of the program will excel in team work, ethics, and communication skills and contribute to the benefit to the society.

Program Outcomes:

PO - 1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO - 2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO - 3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO - 4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO - 5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO - 6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO - 7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO - 8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO - 9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO - 10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO - 11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO - 12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes

PSO - I: Apply the Knowledge of Computing Skills in building the Software Systems that meet the requirements of Industry and Society.

PSO - II: Apply the Knowledge of Data Engineering and Communication Technologies for Developing Applications in the Domain of Smart and Intelligent Computing.

Syllabus

Course Code	20SA8651	**Year**	III	**Semester**	II
Course Category	SOC	**Branch**	CSE	**Course Type**	Practical
Credits	2	**L-T-P**	1-0-2	**Prerequisites**	Programming with Java, DBMS, Advanced Java and Web Technologies
Continuous Internal Evaluation:	--	**Semester End Evaluation:**	50	**Total Marks:**	50

Course Outcomes		
Upon successful completion of the course, the student will be able to		
CO1	Apply the basic of android to develop android applications	**L3**
CO2	Develop various applications as an individual or team	**L3**
CO3	Develop an effective report based on various programs implemented	**L3**
CO4	Apply technical knowledge for a given problem and express with an effective oral communication	**L3**
CO5	Analyse outputs generated using android application	**L4**

Contribution of Course Outcomes towards achievement of Program Outcomes & Strength of correlations (3: Substantial, 2: Moderate, 1: Slight)

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1													3	
CO2					2				2					
CO3										3				
CO4										3				
CO5		2												

Course Content		
Expt. No.1	Build mobile application based on the concept activity life cycle with Custom Toast.	CO1, CO2, CO3, CO4, CO5
Expt. No.2	Build mobile application using different layouts (use any 3 layouts)	CO1, CO2, CO3, CO4, CO5
Expt. No.3	Build mobile application using different dialogs (use any 2 dialogs)	CO1, CO2, CO3, CO4, CO5
Expt. No.4	Build mobile application using Recycler View	CO1, CO2, CO3, CO4, CO5
Expt. No.5	Build mobile application to switch from one activity to another using Intent.	CO1, CO2, CO3, CO4, CO5
Expt. No.6	Build mobile application to demonstrate Dynamic Fragments	CO1, CO2, CO3, CO4, CO5
Expt. No.7	Build mobile application server less database SQLite Database, Firebase (cloud-hosted database)	CO1, CO2, CO3, CO4, CO5
Expt. No.8	Build mobile application based on the Google Maps	CO1, CO2, CO3, CO4, CO5

Learning Resources

References:

1. Professional Android, Reto Meier, Ian Lake, Fourth Edition, 2018, Wrox
2. Head First Android Development: A Brain-Friendly Guide, Dawn Griffiths, David Griffiths, 2015, O′Reilly

PVP20 Regulations -

As per 9.3.1(b), the job oriented skill courses may be registered at the college or at any accredited external agency. A student shall submit a record/report on the on the list skills learned. If the student completes job oriented skill course at external agency, a certificate from the agency shall be included in the report. The course will be evaluated at the end of the semester for 50 marks (record: 15 marks and viva-voce: 35 marks) along with laboratory end examinations in the presence of external and internal examiner (course instructor or mentor). There are no internal marks for the job oriented skill courses.

Distribution of Marks

S.No.	Criterion	Marks
1	Record	15
2	viva-voce	35

Rubrics

End Examination				**Record**
Procedure	**Execution**	**Viva-voce**	**Output**	
Apply (10)	Individual Performance (10M)	Communication (10M)	Analysis (5M)	**Develop an effective report (15M)**

ANNEXURE

EXPERIMENT-1

AIM:

To Build Mobile Application based on the concept Activity Life Cycle with Custom Toast.

DESCRIPTION:

- Android Toast can be used to display information for the short period of time.
- A toast contains message to be displayed quickly and disappears after sometime.
- The android.widget.Toast class is the subclass of java.lang.Object class.

Toast class

- Toast class is used to show notification for a particular interval of time. After sometime it disappears. It doesn't block the user interaction.

Constants of Toast class

There are only 2 constants of Toast class which are given below:

Constants	Description
public static final int LENGTH_LONG	Displays view for the long duration of time
public static final int LENGTH_SHORT	Displays view for the short duration of time

Methods of Toast Class:

The widely used methods of Toast class are given below.

Method	Description
public static Toast makeText(Context context, CharSequence text, int duration)	Makes the toast containing text and duration
public void show()	Displays toast

Create Android Application:

To create a “Hello World” android application user need to follow the below steps

1. Creation of new Android project

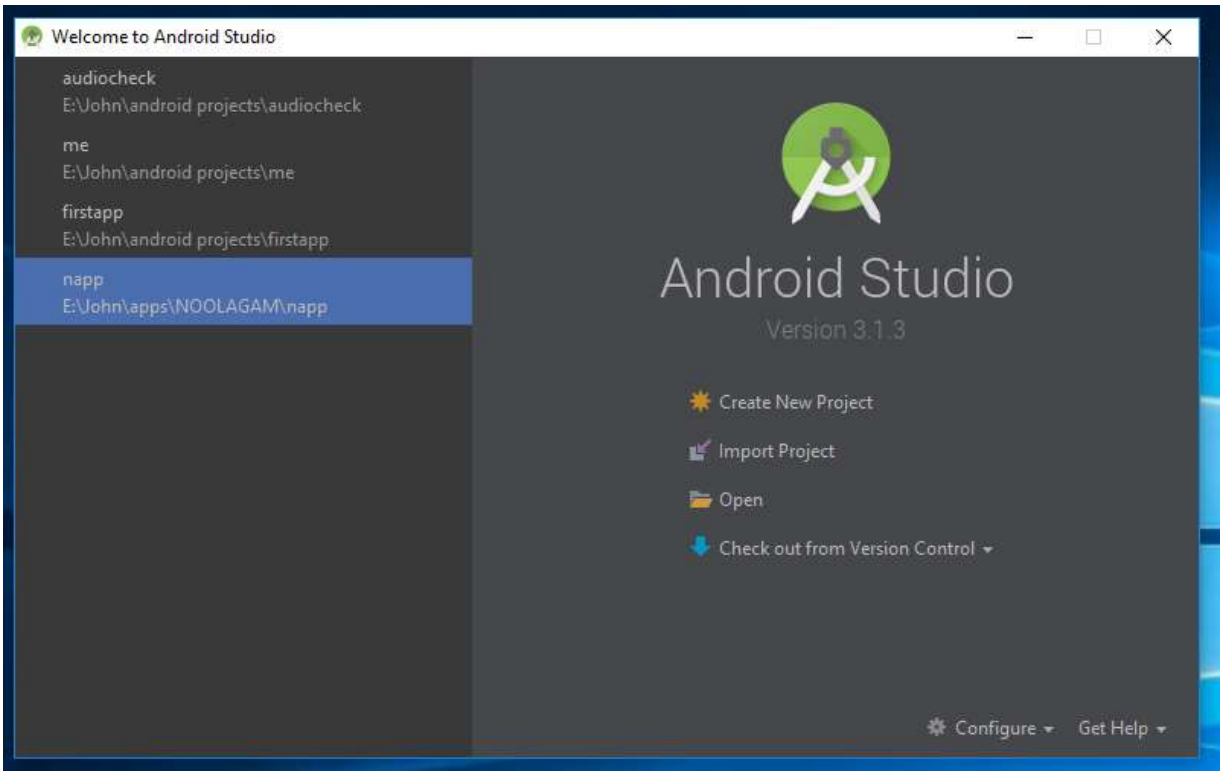
2. Write the message

3. Run the Android application

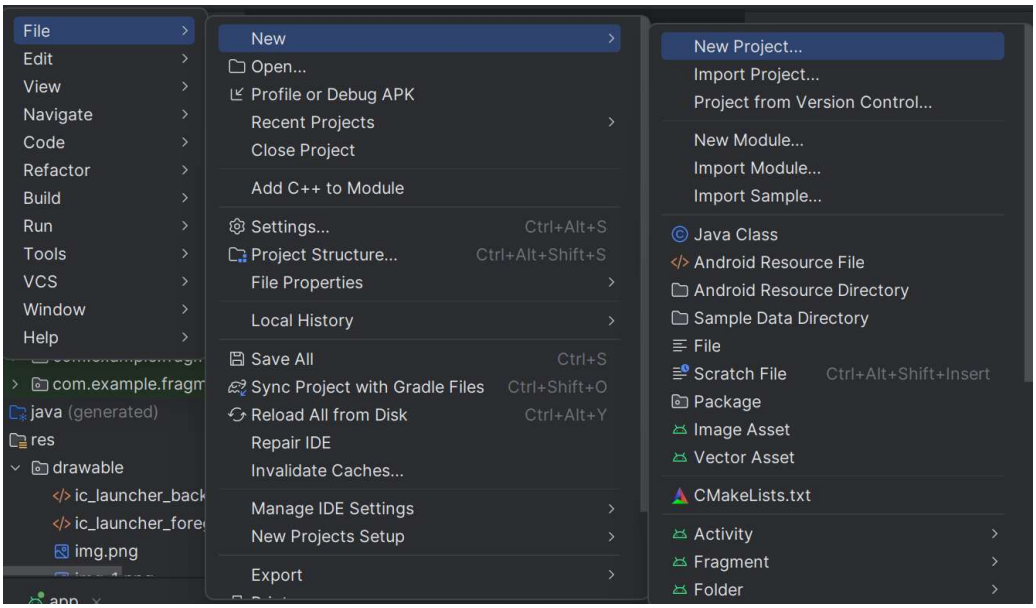
Steps to create a new project in Android Studio:

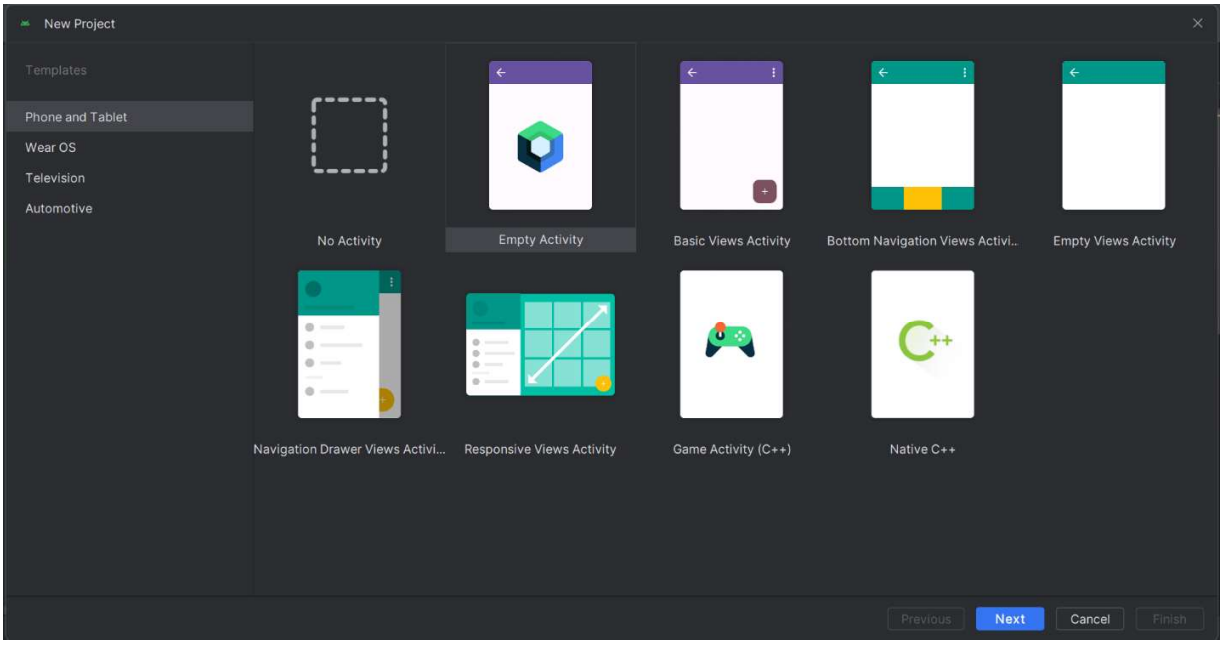
In order to create new Android studio project:

1. Go to start a new Android Project



2.Select New Project

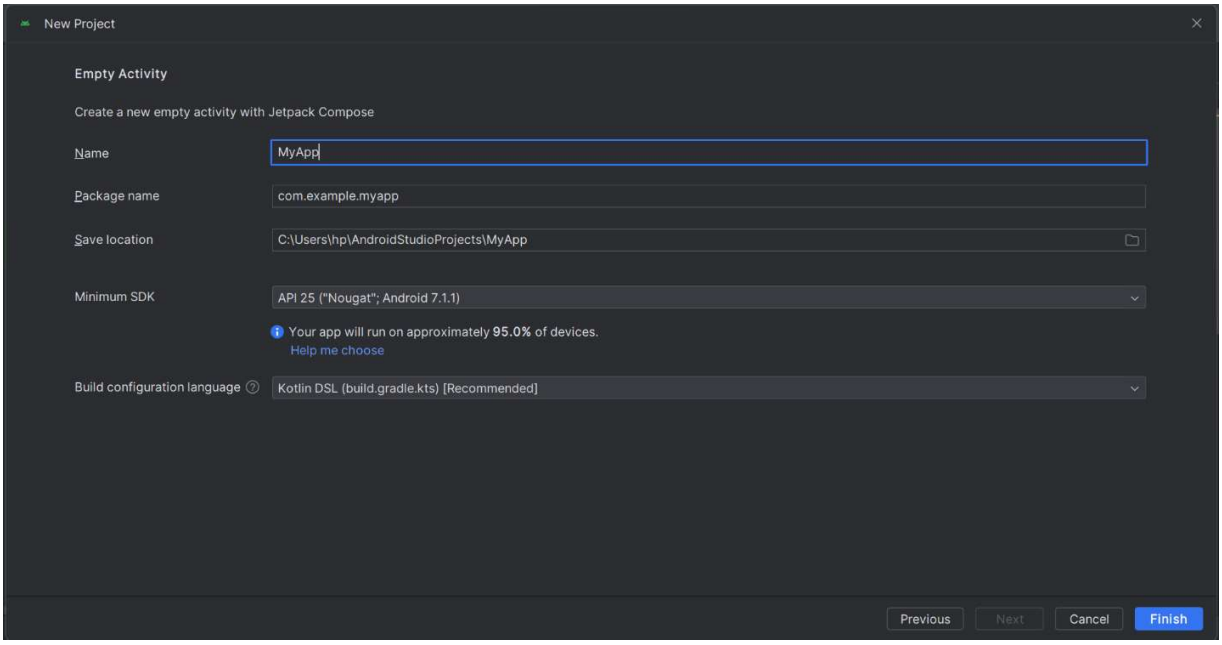




3.Select the Type of Activity

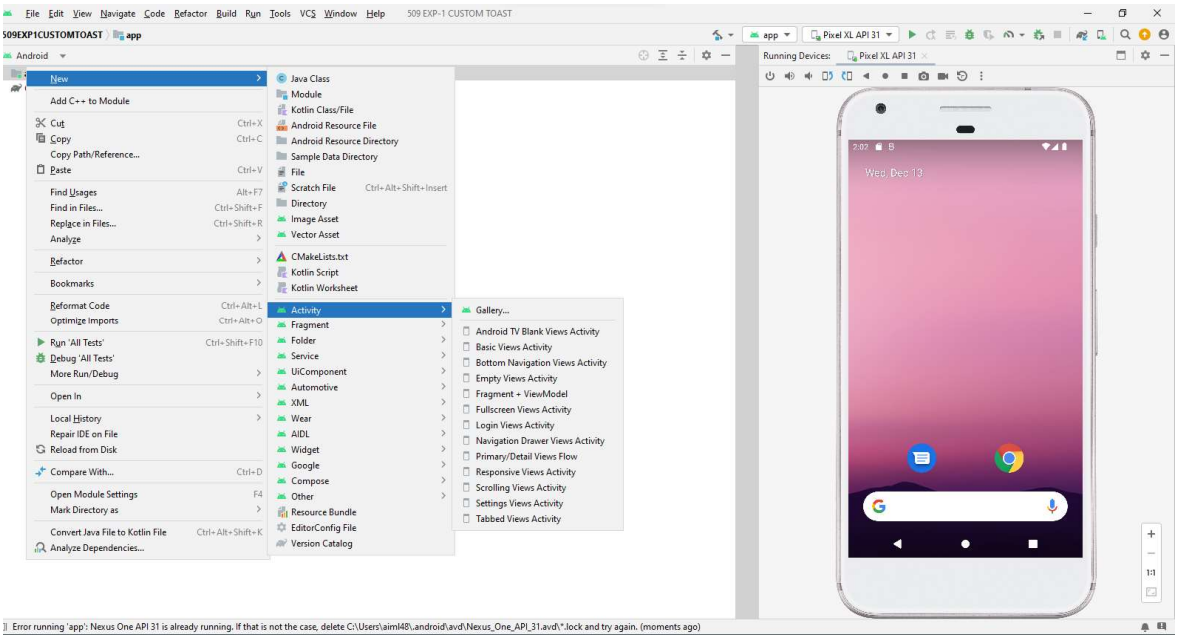
4.Write The below data

- Application name
- Company Domain
- Project location and package name of application

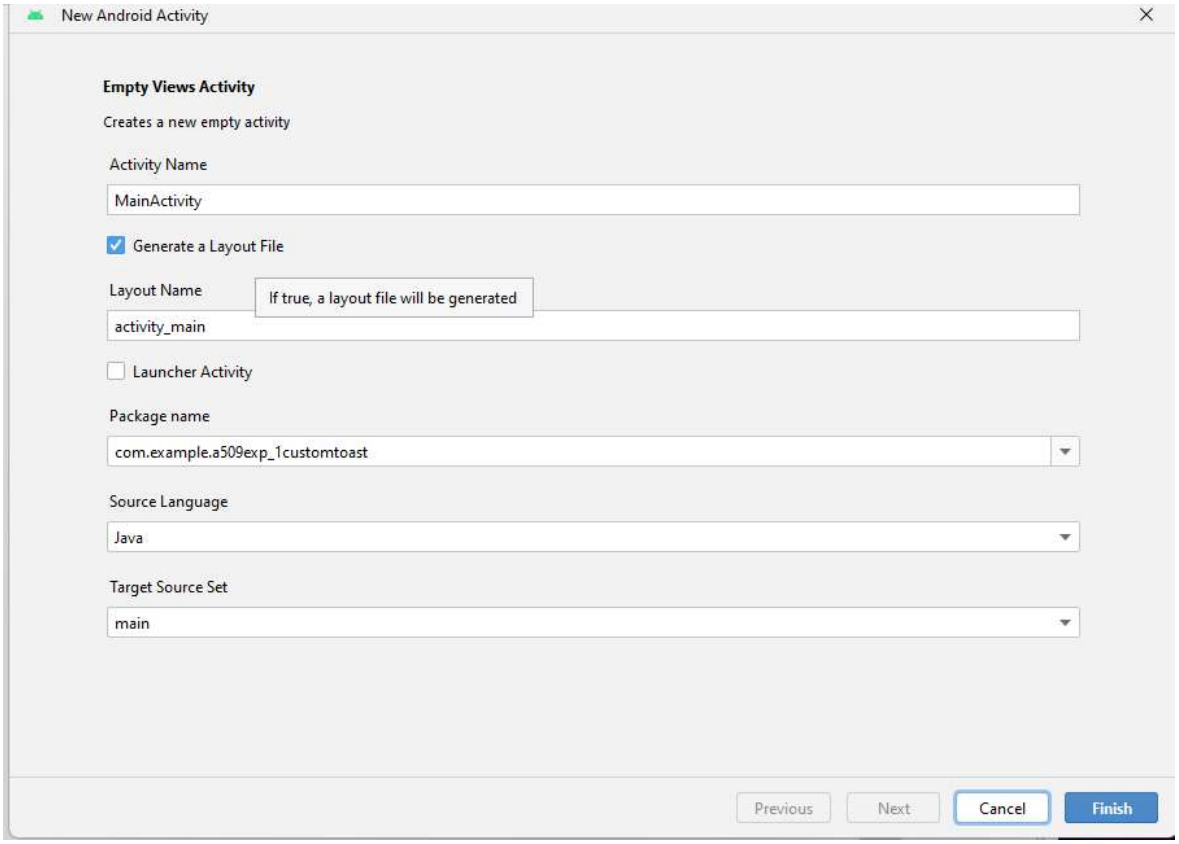


- Select the API level of application. Then click next

5. Create an empty activity



6. Write the activity name. And click on finish ,with this the Android studio will generate the activity class and required configuration files . With this an android project will be created.



TOAST

activity_toast_c0.xml:

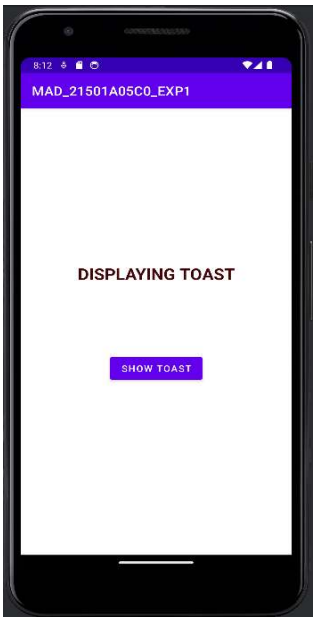
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".toast_c0">
<Button
      android:id="@+id/bid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="display_toast"
      android:text="SHOW TOAST"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
```

```
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.588" />
<TextView
      android:id="@+id/tid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="DISPLAYING TOAST"
android:textColor="#420505"
android:textSize="25sp"
android:textStyle="bold"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.364" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

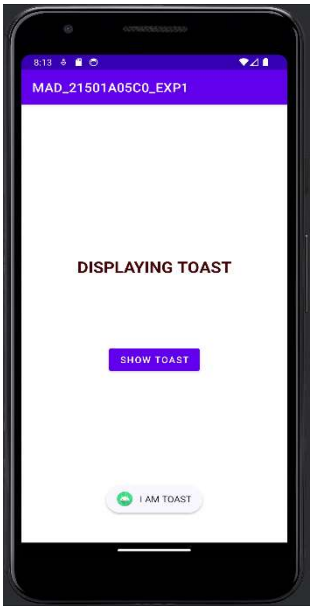
toast_c0.java:

```
package com.example.mad_21501a05c0;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
public class toast_c0 extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_toast_c0);
   }
public void display_toast(View v)
   {
Toast.makeText(this, "I AM TOAST",Toast.LENGTH_LONG).show();
   }
}
```

OUTPUT:



INITIAL DISPLAY



ON CLICK OF **SHOW TOAST**

CUSTOM TOAST:

activity_custom_toast.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".custom_toast">
<Button
      android:id="@+id/bid3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="display_customtoast"
      android:text="SHOW CUSTOM TOAST"
android:textSize="21sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.496"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.477" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

new_layout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
   android:id="@+id/lid">
<TextView
      android:id="@+id/tid2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="HIE,I AM CUSTOM TOAST"
android:textColor="#AA1308"
android:textSize="25sp"
android:textStyle="bold" />
</LinearLayout>
```

custom_toast.java:

```
package com.example.mad_21501a05c0;
import androidx.appcompat.app.AppCompatActivity;
import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.Toast;
public class custom_toast extends AppCompatActivity {
   View cv;
   @SuppressLint("MissingInflatedId")
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_custom_toast);
       LayoutInflater l=this.getLayoutInflater();
cv=l.inflate(R.layout.new_layout,findViewById(R.id.lid));
   }
public void display_customtoast(View v)
   {
       Toast t=new Toast(this);
t.setDuration(Toast.LENGTH_LONG);
t.setView(cv);
t.show();
   }
}
```

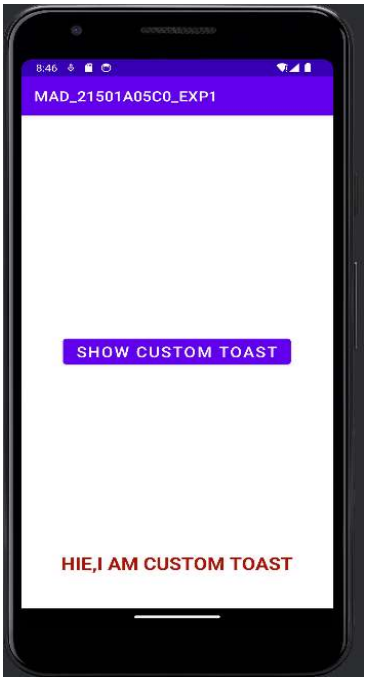
AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
       android:dataExtractionRules="@xml/data_extraction_rules"
       android:fullBackupContent="@xml/backup_rules"
       android:icon="@mipmap/ic_launcher"
android:label="MAD_21501A05C0_EXP1"
       android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
       android:theme="@style/Theme.MAD_21501A05C0"
tools:targetApi="31">
<activity
android:name=".custom_toast"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
```

```
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<activity
        android:name=".toast_c0"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:

INITIAL DISPLAY



ON CLICK OF **SHOW CUSTOM TOAST**

EXPERIMENT-2

AIM:

Build mobile application using different layouts (use any 3 Layouts).

DESCRIPTION:

Linear Layout:

Linear layout is a simple layout used in android for layout designing.

- In the Linear layout all the elements are displayed in linear fashion means all the child elements of a linear layout are displayed according to its orientation.
- The value for orientation property can be either horizontal or vertical.

Types Of Linear Layout Orientation: There are two types of linear layout orientation:

1. Vertical

2. Horizontal

As the name specified these two orientations are used to arrange there child one after the other, in a line, either vertically or horizontally. Let's we describe these in detail.

1.Vertical:

In this all the child are arranged vertically in a line one after the other. In below code snippets we have specified orientation "vertical" so the child views of this layout are displayed vertically.

2.Horizontal:

In this all the child are arranged horizontally in a line one after the other. In below code snippets we have specified orientation "horizontal" so the child views of this layout are displayed horizontally.

Property/Attribute	Description
android:layout_width	It specifies the width of the View or View Group
android:layout_height	It specifies the height of the View or View Group
android:layout_mariginTop	It specifies extra space on the top side of View or View Group
android:layout_marginBottom	It specifies extra space on the bottom side of View or View Group
android:layout_mariginLeft	It specifies extra space on the left side of View or View Group
android:layout_mariginRight	It specifies extra space on the right side of View or View Group

Frame Layout:

- Frame Layout is designed to block out an area on the screen to display a single item.
- However, add multiple children to a Frame Layout and control their position within the Frame Layout by assigning gravity to each child, using the **android:layout_gravity** attribute.
- Child views are drawn in a **stack**, with the most recently added child on top.
- The size of the Frame Layout is the **size of its largest child** (plus padding).

Table Layout

- A layout that arranges its children into rows and columns.
- A TableLayout consists of a number of TableRowobjects, each defining a row
- Do not display border lines for their rows, columns, or cells.
- Each row has zero or more cells;
- Each cell can hold one View object.
- Width(of a child) is always MATCH_PARENT.
- We can specify certain columns as shrinkable or stretchable by calling setColumnShrinkable() or setColumnStretchable()
- stretchable **-**to fit any extra space
- shrinkable - shrunk to fit the table into its parent object

5.2.1:Relative Layout:

- The Relative Layout is very flexible layout used in android for custom layout designing.
- RelativeLayout is a view group that displays child views in relative positions.
- The position of each view can be specified as relative to sibling elements (such as to the left-of or below another view) or in positions relative to the parent RelativeLayout area (such as aligned to the bottom, left or center).

Property/Attribute	Description
android:layout_alignParentBottom	It places the bottom of the element on the bottom of the container
android:layout_alignParentLeft	It places the left of the element on the left side of the container
android:layout_alignParentRight	It places the right of the element on the right side of the container
android:layout_alignParentTop	It places theelement at the top of the container
android:layout_centerHorizontal	It ceneters the element horizontally with in its Parent Container
android:layout_centerVertical	It ceneters the element vertically with in its Parent Container

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">
<TextView
      android:id="@+id/tid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="LAYOUT ACTIVITY"
android:textSize="35sp"
android:textStyle="bold"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.125" />
<Button
      android:id="@+id/bm1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="button_tlayout"
      android:text="TABLE LAYOUT"
android:textSize="25sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.331"
app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/tid1"
app:layout_constraintVertical_bias="0.377" />
<Button
      android:id="@+id/button3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginBottom="72dp"
android:onClick="button_linear_layout"
      android:text="LINEAR LAYOUT"
      app:layout_constraintBottom_toTopOf="@+id/bm1"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.227"
```

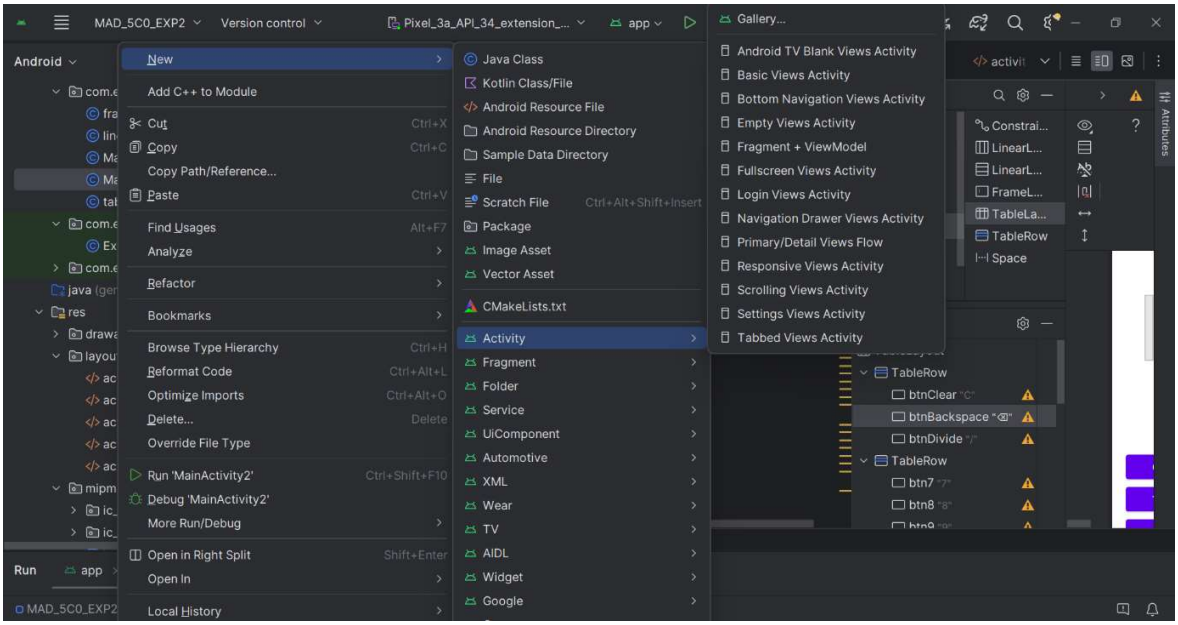
```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="1.0" />
<Button
      android:id="@+id/button2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="button_framelayout"
      android:text="FRAME LAYOUT"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.208"
app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/bm1"
app:layout_constraintVertical_bias="0.297" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

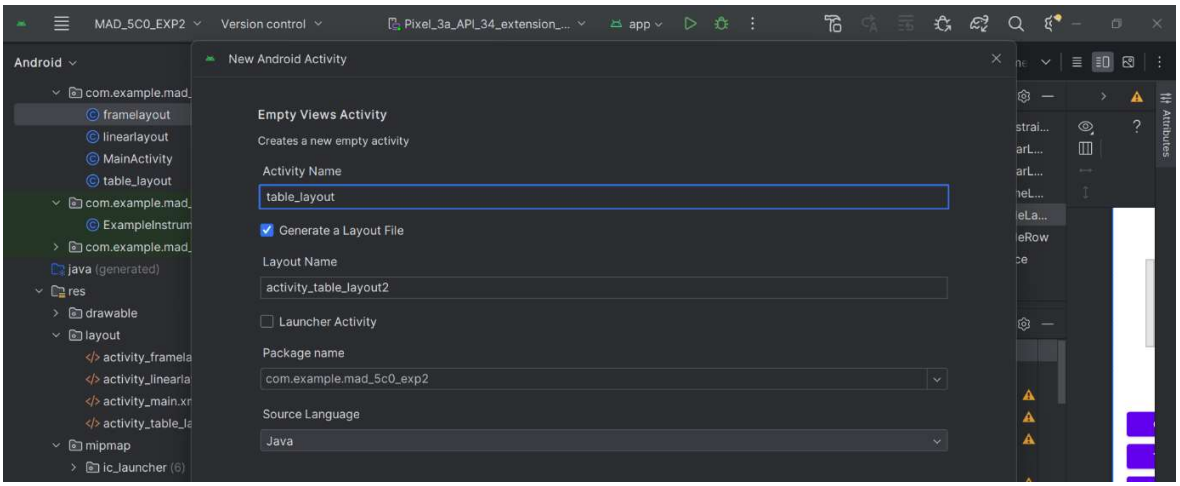
```
package com.example.mad_5c0_exp2;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class MainActivity extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
   }
public void button_tlayout(View v)
   {
      Intent i=new Intent(this,table_layout.class);
this.startActivity(i);
   }
public void button_linear_layout(View v)
   {
      Intent i=new Intent(this,linearlayout.class);
this.startActivity(i);
   }
public void button_framelayout(View v)
   {
      Intent i=new Intent(this,framelayout.class);
this.startActivity(i);
   }
}
```

TABLELAYOUT:

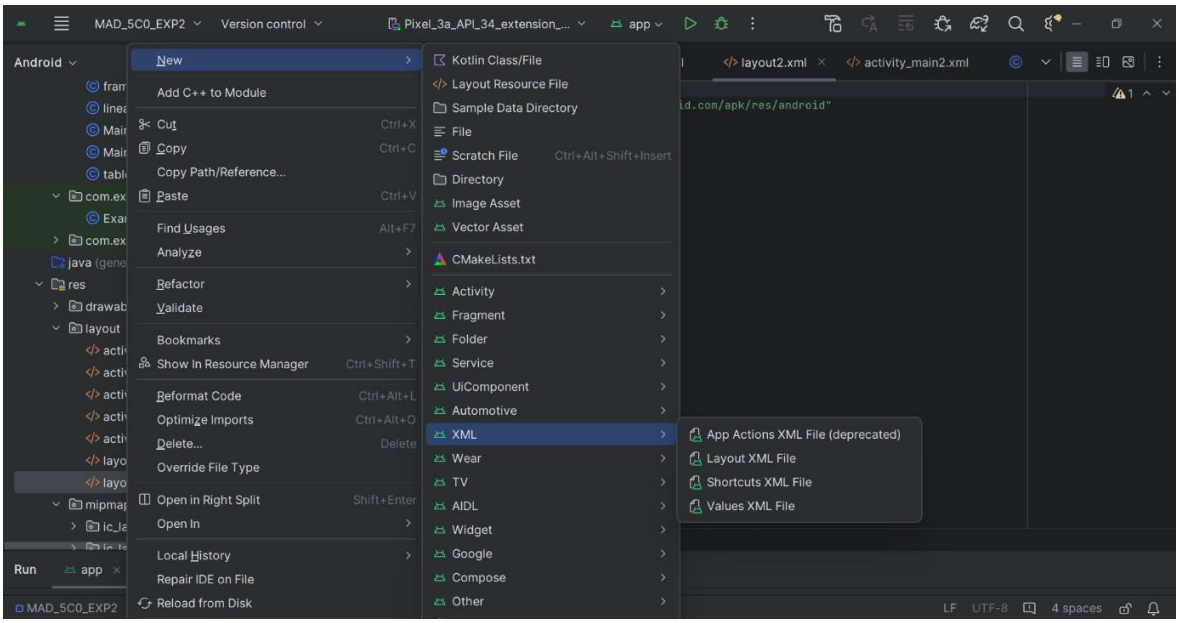
- Creation of new Activity

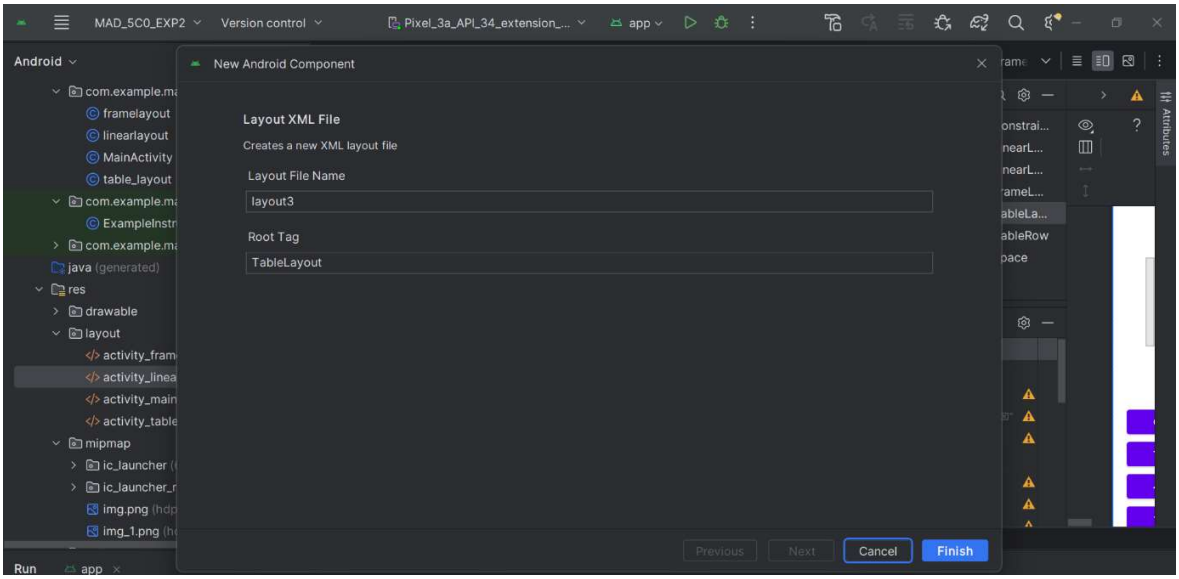


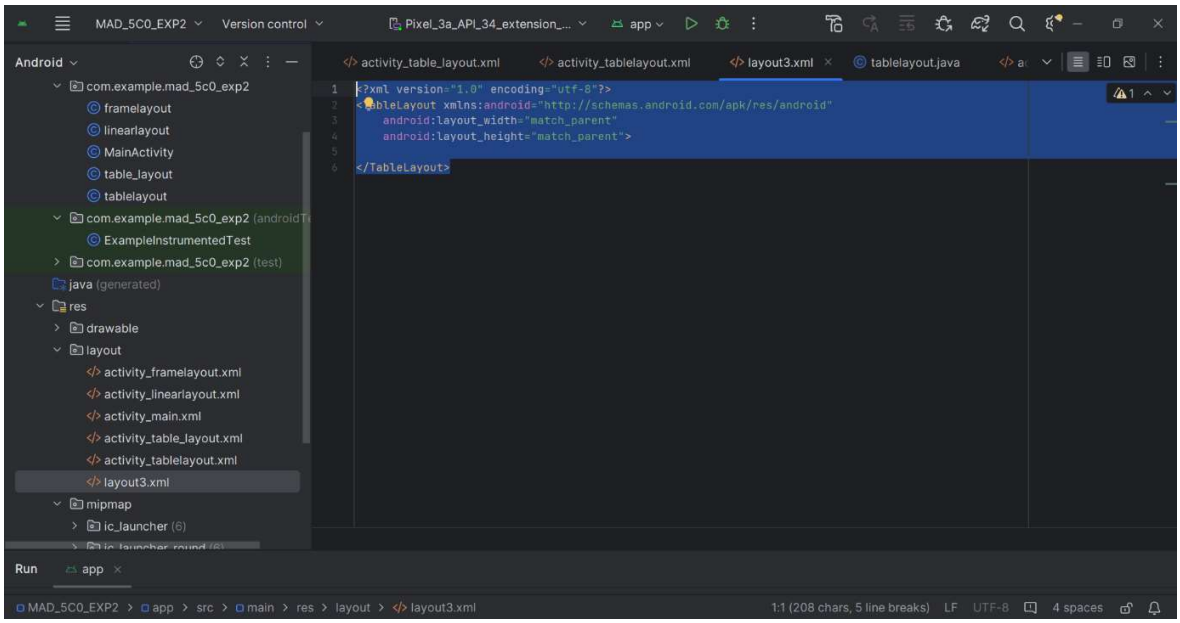
- Disable the Launcher Activity



- Creation of new xml file with required Root Tag





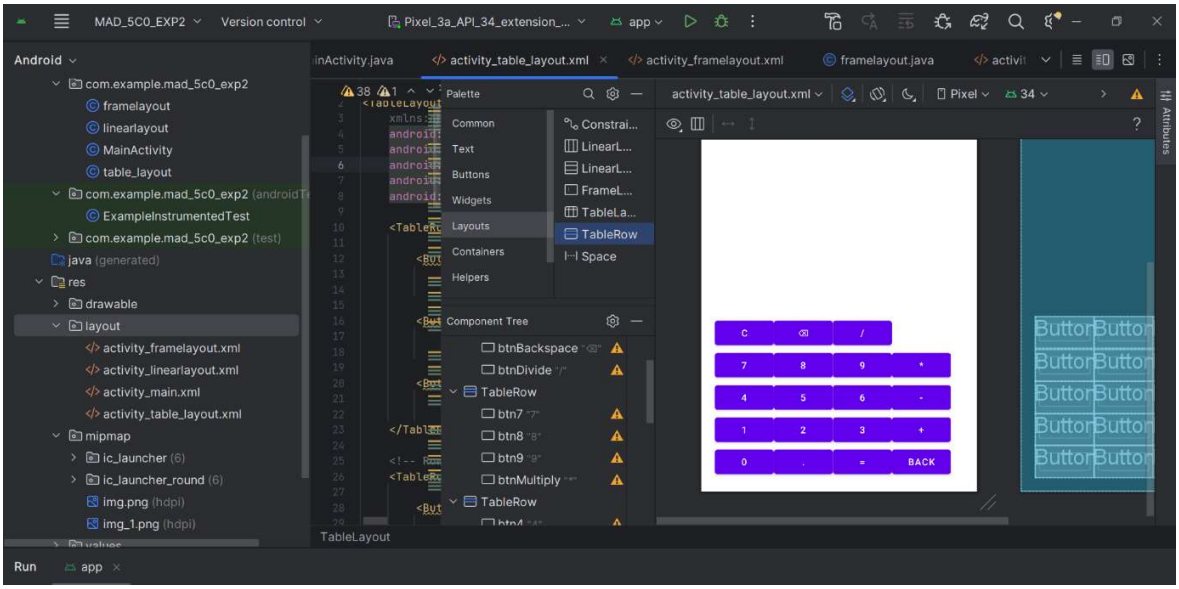


- Copy the structure of the requried layout and make necessary changes in the Activity created above

activity_table_layout.xml:

- Creation of TableRow and adding required components to it by drag and drop.

<?xml version="1.0" encoding="utf-8"?>



```
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:gravity="center|bottom"
android:orientation="horizontal"
android:padding="20sp">
<TableRow>
<Button
        android:id="@+id/btnClear"
        android:text="C" />
<Button
        android:id="@+id/btnBackspace"
        android:text="□" />
<Button
        android:id="@+id/btnDivide"
        android:text="/" />
</TableRow>
<!-- Row 2 -->
<TableRow>
<Button
        android:id="@+id/btn7"
        android:text="7" />
<Button
        android:id="@+id/btn8"
        android:text="8" />
<Button
        android:id="@+id/btn9"
        android:text="9" />
<Button
        android:id="@+id/btnMultiply"
        android:text="*" />
</TableRow>
<!-- Row 3 -->
<TableRow>
<Button
        android:id="@+id/btn4"
        android:text="4" />
<Button
        android:id="@+id/btn5"
        android:text="5" />
<Button
        android:id="@+id/btn6"
        android:text="6" />
<Button
        android:id="@+id/btnSubtract   android:text="-" /></TableRow>
```

```
<!-- Row 4 -->
<TableRow>
<Button
        android:id="@+id/btn1"
        android:text="1" />
<Button
        android:id="@+id/btn2"
        android:text="2" />
<Button
        android:id="@+id/btn3"
        android:text="3" />
<Button
        android:id="@+id/btnAdd"
        android:text="+" />
</TableRow>
<!-- Row 5 -->
<TableRow>
<Button
        android:id="@+id/btn0"
        android:text="0" />
<Button
        android:id="@+id/btnDot"
        android:text="." />
<Button
        android:id="@+id/btnEquals"
        android:text="=" />
<Button
        android:id="@+id/bid43"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="switch_backtl"
        android:text="BACK" />
</TableRow>
</TableLayout>
```

table_layout.java:

```
package com.example.mad_5c0_exp2;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;


public class table_layout extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_table_layout);
   }
public void switch_backtl(View v)
   {
      Intent i=new Intent(this,MainActivity.class);
this.startActivity(i);
   }
}
```

activity_linearlayout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/ll"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:orientation="horizontal">
<Button
      android:id="@+id/button"
android:layout_width="match_parent"
android:layout_height="wrap_content"
      android:text="Button" />
<Button
      android:id="@+id/button8"
android:layout_width="match_parent"
android:layout_height="wrap_content"
      android:text="Button" />
<Button
      android:id="@+id/button9"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:onClick="switch_backll"
      android:text="back" />
<Button
      android:id="@+id/bid21"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:onClick="changeorientation"
      android:text="change orientation" />
</LinearLayout>
```

linearlayout.java:

```
package com.example.mad_5c0_exp2;
```

```
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.LinearLayout;
public class linearlayout extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_linearlayout);
   }
public void changeorientation(View v)
   {
      LinearLayout ll;
ll=findViewById(R.id.ll);
if(ll.getOrientation()==LinearLayout.HORIZONTAL) {
ll.setOrientation(LinearLayout.VERTICAL);
      }
      else {
ll.setOrientation(LinearLayout.HORIZONTAL);
      }}
public void switch_backll(View v)
      {
         Intent i=new Intent(this,MainActivity.class);
this.startActivity(i);
      }
   }
```

activity_framelayout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
android:layout_width="409dp"
android:layout_height="729dp"
tools:layout_editor_absoluteX="1dp"
tools:layout_editor_absoluteY="1dp">
<ImageView
      android:id="@+id/imageView4"
android:layout_width="match_parent"
android:layout_height="782dp"
      app:srcCompat="@mipmap/img_1" />
<TextView
      android:id="@+id/textView"
android:layout_width="424dp"
```

```
android:layout_height="50dp"
      android:text="LOVE YOU DEAR AIRFORCE"
android:textAlignment="center"
android:textColor="#000000"
android:textSize="30sp"
android:textStyle="bold" />
</FrameLayout>
```

framelayout.java:

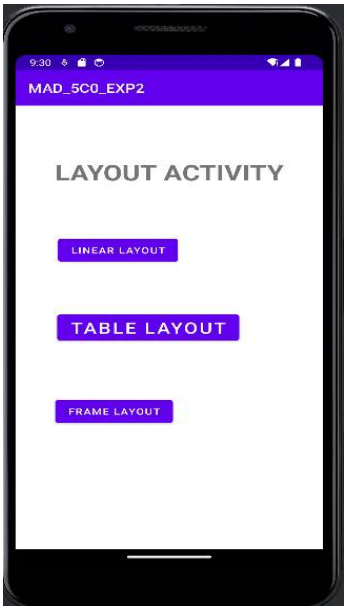
```
package com.example.mad_5c0_exp2;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
public class framelayout extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_framelayout);
   }
}
```

AndroidManifest.xml:

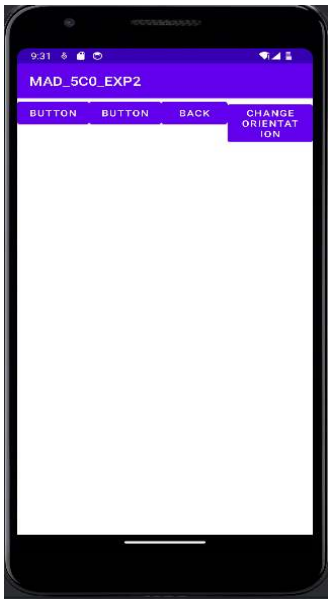
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
      android:dataExtractionRules="@xml/data_extraction_rules"
      android:fullBackupContent="@xml/backup_rules"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
      android:theme="@style/Theme.MAD_5C0_EXP2"
tools:targetApi="31">
<activity
android:name=".framelayout"
android:exported="false" />
<activity
android:name=".linearlayout"
android:exported="false" />

<activity
android:name=".table_layout"
android:exported="false" />
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:

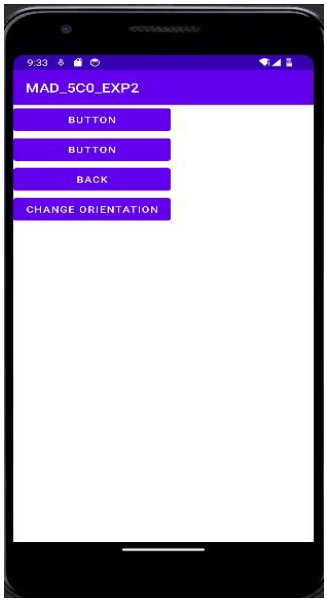


INITIAL DISPLAY

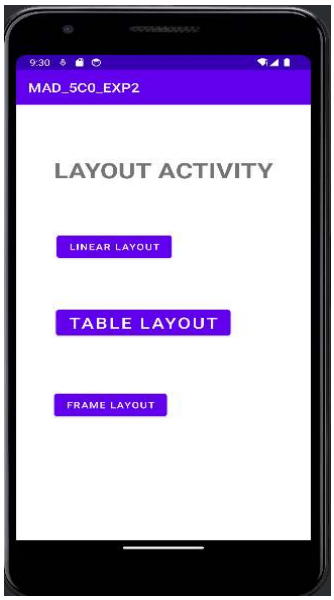


ON CLICK OF **LINEAR LAYOUT**

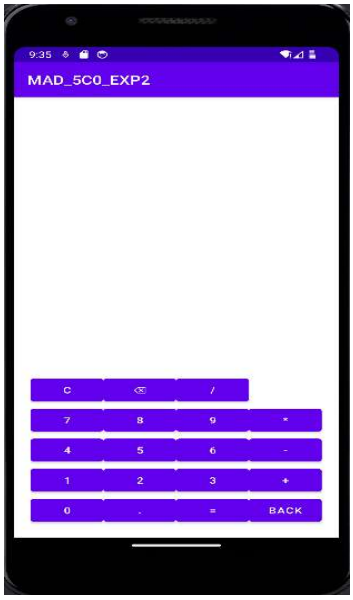
HORIZONTAL ORIENTATION



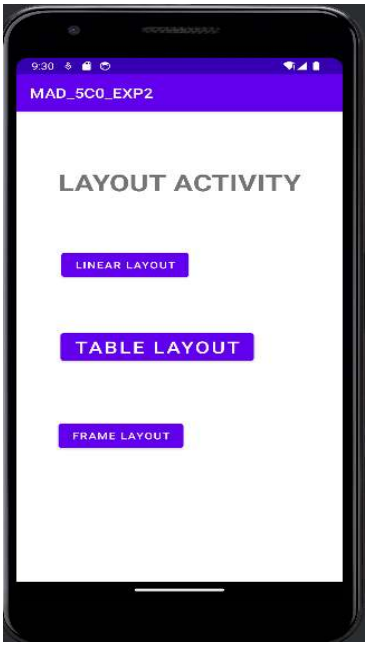
ON CLICK **OF CHANGE ORIENTATION**



ON CLICK OF **BACK**



ON CLICK OF **TABLE LAYOUT**



ON CLICK OF **BACK**

ON CLICK OF **FRAME LAYOUT**

EXPERIMENT-3

AIM:

Build mobile application using different dialogs (use any 2 dialogs).

DESCRIPTION:

Date Picker:

- In Android, DatePicker is a widget used to select a date. It allows to select date by day, month and year in your custom UI (user interface).
- If we need to show this view as a dialog then we have to use a DatePickerDialog class.

Property/Attribute	Description
getDayOfMonth()	This method gets the selected day of month
getMonth()	This method gets the selected month
getYear()	This method gets the selected year
setMaxDate(long maxDate)	This method sets the maximal date supported by this DatePicker in milliseconds since January 1, 1970 00:00:00 in getDefault() time zone
setMinDate(long minDate)	This method sets the minimal date supported by this NumberPicker in milliseconds since January 1, 1970 00:00:00 in getDefault() time zone
getCalendarView()	getCalendarView()
updateDate(int year, int month, int dayOfMonth)	This method updates the current date
getFirstDayOfWeek()	This Method returns first day of the week

Time Picker:

- In Android, TimePicker is a widget used for selecting the time of the day in either AM/PM mode or 24 hours mode.
- The displayed time consist of hours, minutes and clock format. If we need to show this view as a Dialog then we have to use a TimePickerDialog class.
- In order to get the time selected by the user on the screen, you will use getCurrentHour() and getCurrentMinute() method of the TimePicker Class. Their syntax is given below.

```
int hour = timePicker1.getCurrentHour();

int min = timePicker1.getCurrentMinute();
```

Apart from these methods, there are other methods in the API that gives more control over TimePicker Component. They are listed below:

Property/Attribute	Description
is24HourView()	This method returns true if this is in 24 hour view else false
isEnabled()	This method returns the enabled status for this view
setCurrentHour(Integer currentHour)	This method sets the current hour
setCurrentMinute(Integer currentMinute)	This method sets the current minute
setEnabled(boolean enabled)	This method set the enabled state of this view
setIs24HourView(Boolean is24HourView)	This method set whether in 24 hour or AM/PM mode
setOnTimeChangedListener(TimePicker. OnTimeChangedListener onTimeChangedListener)	This method Set the callback that indicates the time has been adjusted by the user

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:onClick="btn_timepicker"
tools:context=".MainActivity">
<Button
      android:id="@+id/bid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="btn_showdialog"
      android:text="SHOW DIALOG"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.285" />
<Button
      android:id="@+id/bid2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="btn_datepicker"
      android:text="DATE PICKER"
      app:layout_constraintBottom_toTopOf="@+id/bid3"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/bid1"
app:layout_constraintVertical_bias="0.489" />
<Button
      android:id="@+id/bid3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginBottom="228dp"
android:onClick="btn_timepicker"
      android:text="TIME PICKER"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent" />
<TextView
      android:id="@+id/tidtv"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="DIALOG APPLICATION"
android:textColor="#00BCD4"
android:textColorHint="#6E0000"
android:textSize="34sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.546"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.119" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Mainactivity.java:

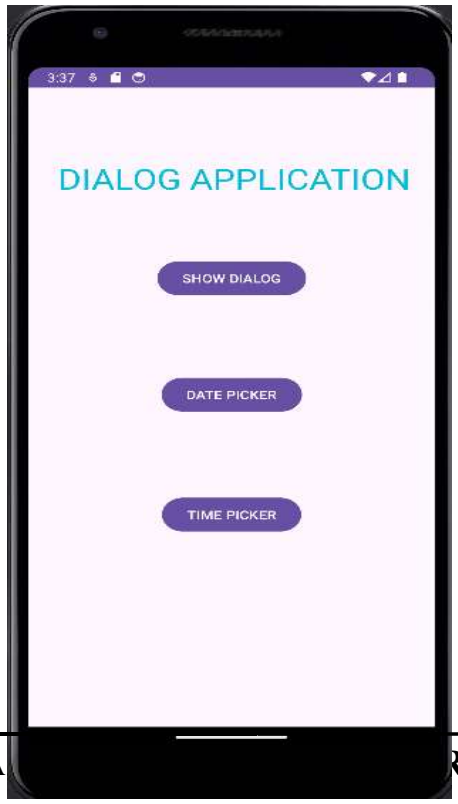
```
package com.example.a21501a05c0_dialogs;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.DatePicker;
import android.widget.TimePicker;
import android.widget.Toast;
import java.time.Month;
import java.time.Year;
import java.util.Calendar;
import java.util.Date;
public class MainActivity extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
   }
public void btn_timepicker(View v) {
      int dh, dmin;
      Calendar c = Calendar.getInstance();
dh = c.get(Calendar.HOUR);
dmin = c.get(Calendar.MINUTE);
      TimePickerDialog tpg = new TimePickerDialog(this, new
TimePickerDialog.OnTimeSetListener() {
         @Override
public void onTimeSet(TimePicker timepicker, int i, int i1) {
```

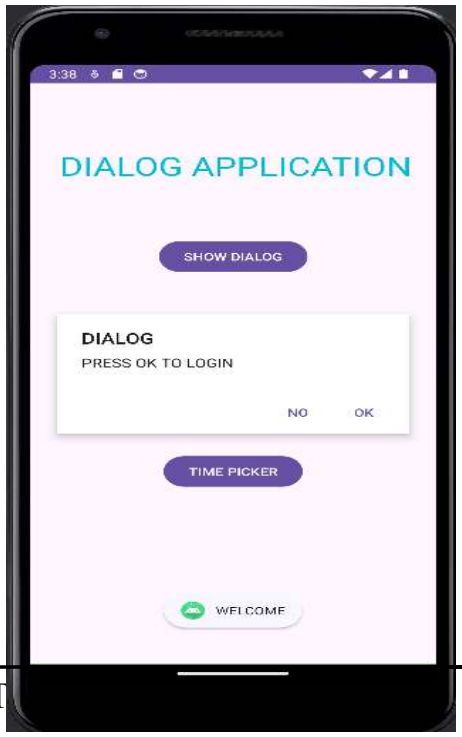
```
Toast.makeText(MainActivity.this, i + "Hr:" + i + "min", Toast.LENGTH_SHORT).show();
            }
        }, dh, dmin, false);
tpg.show();
    }
public void btn_datepicker(View v) {
        int date, month, year;
        Calendar c = Calendar.getInstance();
date = c.get(Calendar.DAY_OF_MONTH);
month = c.get(Calendar.MONTH);
year = c.get(Calendar.YEAR);
        DatePickerDialog dpg = new DatePickerDialog(this, new
DatePickerDialog.OnDateSetListener() {
            @Override
public void onDateSet(DatePicker view, int year, int month, int date) {
Toast.makeText(MainActivity.this, date + "/" + month + "/" + year,
Toast.LENGTH_SHORT).show();
            }
        }, date, month, year);
dpg.setTitle("PICK A DATE");
dpg.setMessage("HELLO THIS IS NC");
dpg.show();
    }
public void btn_showdialog(View v)
    {
        Log.d("NC_DIALOG","WELCOME TO DIALOG 1");
        AlertDialog.Builder abd=new AlertDialog.Builder(this);
        Log.d("NC_DIALOG","WELCOME TO DIALOG 2");
Toast.makeText(this,"WELCOME",Toast.LENGTH_SHORT).show();
abd.setTitle("DIALOG");
abd.setMessage("PRESS OK TO LOGIN");
abd.setCancelable(true);
abd.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            @Override
public void onClick(DialogInterface dialog, int which) {
Toast.makeText(MainActivity.this,"CLICKED ON OK",Toast.LENGTH_SHORT).show();
            }
        });
abd.setNegativeButton("NO", new DialogInterface.OnClickListener() {
            @Override
public void onClick(DialogInterface dialog, int which) {
Toast.makeText(MainActivity.this,"CLICKED ON NO",Toast.LENGTH_SHORT).show();
            }
        });
        AlertDialog ad=abd.create();
        Log.d("NC_DIALOG","WELCOME TO DIALOG 3");
```

```
ad.show();
    }
}
```

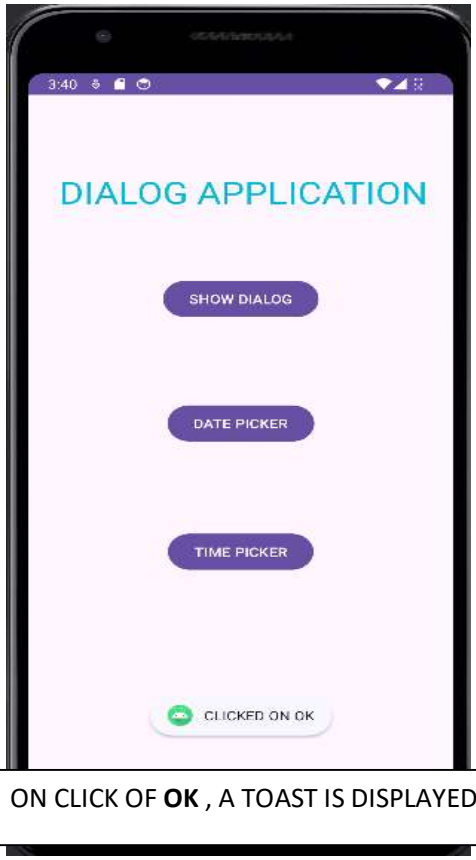
AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
android:label="_21501A05C0_Dialogs"
        android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
        android:theme="@style/Theme._21501A05C0_Dialogs"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```



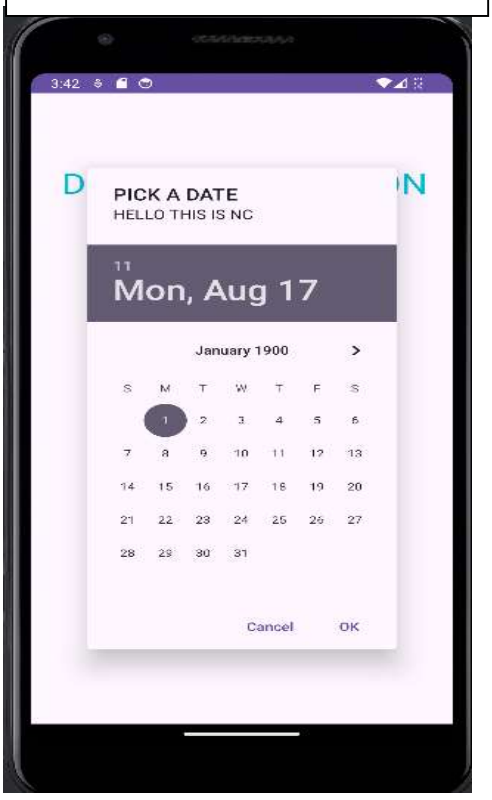


INITIAL DISPLAY

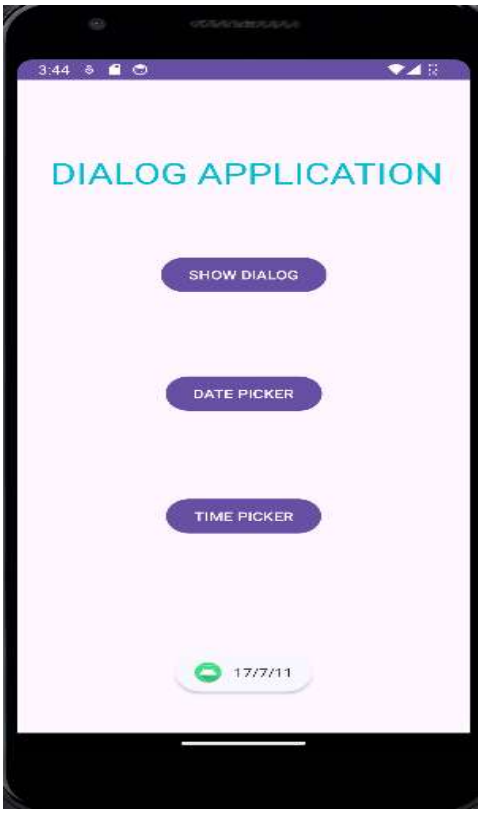


ON CLICK OF **OK** , A TOAST IS DISPLAYED

ON CLICK OF **SHOW DIALOG**



ON CLICK OF **DATE PICKER**



A TOAST DISPLAYED AFTER PICKING DATE

ON CLICK OF **TIME PICKER**

A TOAST DISPLAYED AFTER PICKING TIME

EXPERIMENT-4

AIM:

Build mobile application using Recycler View.

DESCRIPTION:

Recycler View is a View Group added to the android studio as a successor of the Grid View and List View. It is an improvement on both of them and can be found in the latest v-7 support packages. It has been created to make possible construction of any lists with XML layouts as an item which can be customized vastly while improving on the efficiency of ListViews and GridViews. This improvement is achieved by recycling the views which are out of the visibility of the user. For example, if a user scrolled down to a position where items 4 and 5 are visible; items 1, 2, and 3 would be cleared from the memory to reduce memory consumption. Implementation: To implement a basic RecyclerView three sub-parts are needed to be constructed which offer the users the degree of control they require in making varying designs of their choice.

1. The Card Layout: The card layout is an XML layout which will be treated as an item for the list created by the RecyclerView.
2. The ViewHolder: The ViewHolder is a java class that stores the reference to the card layout views that have to be dynamically modified during the execution of the program by a list of data obtained either by online databases or added in some other way.
3. The Data Class: The Data class is a custom java class that acts as a structure for holding the information for every item of the RecyclerView.

The Adapter: The adapter is the main code responsible for RecyclerView. It holds all the important methods dealing with the implementation of RecylcerView. The basic methods for a successful implementation are:

- onCreateViewHolder: which deals with the inflation of the card layout as an item for the RecyclerView.
- onBindViewHolder: which deals with the setting of different data and methods related to clicks on particular items of the RecyclerView.
- getItemCount: which Returns the length of the RecyclerView.
- onAttachedToRecyclerView: which attaches the adapter to the

  RecyclerView.

Steps for implementing RecyclerView:

1. If you're going to use RecyclerView, there are a few things you need to do. They are explained in detail in the following sections.
2. Decide how the list or grid looks. Ordinarily, you can use one of the RecyclerView library's standard layout managers.
3. Design how each element in the list looks and behaves. Based on this design, extend the ViewHolder class. Your version of ViewHolder provides all the functionality for your list items. Your view holder is a wrapper around a View, and that view is managed by RecyclerView.
4. Define the Adapter that associates your data with the ViewHolder views.

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">
<androidx.recyclerview.widget.RecyclerView
      android:id="@+id/rv_students"
android:layout_width="409dp"
android:layout_height="729dp"
tools:ignore="MissingConstraints"
tools:layout_editor_absoluteX="1dp"
tools:layout_editor_absoluteY="1dp" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

```
package com.example.recycler_view_5c0_exp4;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.os.Bundle;
import java.util.ArrayList;
import java.util.List;
public class MainActivity extends AppCompatActivity {
```

```
    RecyclerView rv;
    List<Student> st_list;
    @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
rv=this.findViewById(R.id.rv_students);
rv.setLayoutManager(new LinearLayoutManager(this));
        st_list=new ArrayList<Student>();
        st_list.add(new Student(15567,"NIKHIL CHOWHAN",R.drawable.img_1));
        st_list.add(new Student(12323,"AVANISH",R.drawable.img_2));
        st_list.add(new Student(87124,"KUMAR MOHAN",R.drawable.img_3));
rv.setAdapter(new StudentAdapter(getApplicationContext(),st_list));
    }
}
```

layout.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="wrap_content">
<ImageView
android:layout_marginTop="10dp"
        android:id="@+id/imageView"
android:layout_width="200dp"
android:layout_height="200dp"
        tools:srcCompat="@tools:sample/avatars" />
<TextView
        android:id="@+id/textView_1"
android:layout_width="match_parent"
android:layout_height="match_parent"
        android:layout_toEndOf="@id/imageView"
        android:text="student roll"
android:textSize="30sp" />
<TextView
        android:id="@+id/textView_2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
        android:layout_below="@id/textView_1"
        android:layout_toEndOf="@id/imageView"
        android:text="student name"
```

```
android:textSize="30sp" />
<LinearLayout
android:layout_width="match_parent"
android:layout_height="218dp"
android:layout_marginTop="10dp"
android:orientation="horizontal"></LinearLayout>
</RelativeLayout>
```

StudentViewHolder.java:

```
package com.example.recycler_view_5c0_exp4;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
public class StudentViewHolder extends RecyclerView.ViewHolder
{
   ImageView iv;
   TextView tv_roll,tv_name;
public StudentViewHolder(@NonNull View itemView) {
      super(itemView);
iv=itemView.findViewById(R.id.imageView);
      tv_roll=itemView.findViewById(R.id.textView_1);
      tv_name=itemView.findViewById(R.id.textView_2);
   }
}
```

StudentAdapter.java:

```
package com.example.recycler_view_5c0_exp4;
import android.annotation.SuppressLint;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.ViewGroup;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import java.util.List;
public class StudentAdapter extends RecyclerView.Adapter<StudentViewHolder> {
   Context ct;
   List<Student> student_List;
public StudentAdapter(Context ct, List<Student> student_List) {
      this.ct = ct;
      this.student_List = student_List;
```

```
    }
    @NonNull
    @Override
public StudentViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
return new StudentViewHolder(LayoutInflater.from(ct).inflate(R.layout.layout ,parent,false
));
    }
    @Override
public void onBindViewHolder(@NonNull StudentViewHolder holder, int position) {
    holder.tv_roll.setText(String.valueOf(student_List.get(position).getRoll()));
    holder.tv_name.setText(String.valueOf(student_List.get(position).getName()));
holder.iv.setImageResource(student_List.get(position).getImage());
    }    @Override
public int getItemCount() {
return student_List.size()  ;
    }
}
```

Student.java:

```
package com.example.recycler_view_5c0_exp4;
public class Student {
    int roll;
    String name;
    int image;
public Student(int roll, String name, int image) {
        this.roll = roll;
        this.name = name;
        this.image = image;
    }
public int getRoll() {
        return roll;
    }
public void setRoll(int roll) {
        this.roll = roll;
    }
public String getName() {
        return name;
    }
public void setName(String name) {
        this.name = name;
    }
public int getImage() {
```

```
        return image;
    }
public void setImage(int image) {
        this.image = image;
    }
}
```

AndroidManifest.java:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
        android:theme="@style/Theme.Recycler_view_5C0_exp4"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:


11:43
LTE
15567
NIKHIL
CHOWHAN
12323
AVANISH
87124
KUMAR
MOHAN

EXPERIMENT-5

AIM:

To Build mobile application to switch from one activity to another using Intent.

DESCRIPTION:

Intent:

- Intent is a simple message object that is used to communicate between android components such as
- activities, content providers, broadcast receivers and services. Intents are also used to transfer data between
- activities.
- Intents are used generally for starting a new activity using startActivity().

Use of Intent:

1. For Launching an Activity
2. To start a New Service
3. For Broadcasting Messages
4. To Display a list of contacts in ListView

Types of Intent in Android:

- Intent is of two types
    1. Implicit Intent
    2. Explicit Intent

Implicit Intent:

- Communication between two activities of different application is called implicit intent.
- Intent intent=new Intent(Intent.ACTION_CALL);
  intent.setData(Uri.parse("tel:9177435689"));
  startActivity(intent);

Explicit Intent:

- Communication between two activities of same application is called explicit intent.
- Intent intent=new Intent(this,Activity2.class);
  startActivity(intent);

Switching Between Activities:

- In android, intents are the objects that help a user to navigate from one activity to another activity with in an application.
- The following steps illustrate the process of linking activities using intents.
    1. Create new project named intent using android studio.
    2. Right click on your project package and Select new->Activity->Empty Activity.
    3. Enter the name of the class as Activity2 and then click on finish.

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".main">
<Button
      android:id="@+id/bid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="switch_activity"
      android:text="SWITCH ACTIVITY"
android:textSize="25sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.496"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.573" />
<TextView
      android:id="@+id/tid1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="WELCOME TO ACTIVITY 1"
android:textSize="30sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.492"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.361" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

main_activity.java:

```
package com.example.mad_5c0_exp5;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class main extends AppCompatActivity {
```

```
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
   }
public void switch_activity(View v)
   {
      Intent i=new Intent(this,created.class);
this.startActivity(i);
   }
}
```

activity_created.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".created">
<TextView
      android:id="@+id/tid2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
      android:text="WELCOME TO ACTIVITY 2"
android:textSize="30sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.492"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.393" />
<Button
      android:id="@+id/bid2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:onClick="switch_activity2"
      android:text="BACK TO ACTIVITY 1"
android:textSize="24sp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.498"
```

```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.61" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_created.java:

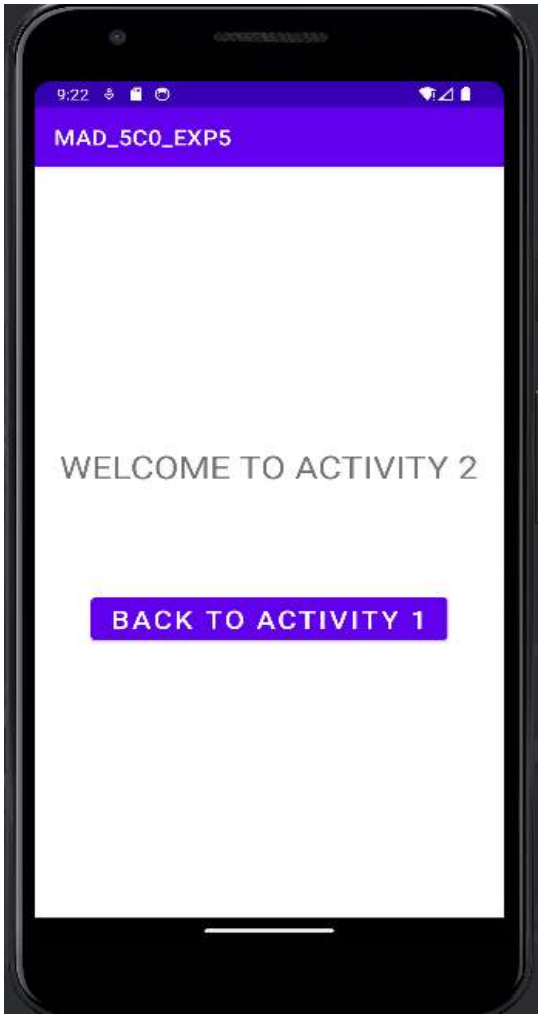
```
package com.example.mad_5c0_exp5;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
public class created extends AppCompatActivity {
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_created);
   }
   public void switch_activity2(View v)
   {
      Intent i=new Intent(this,main.class);
this.startActivity(i);
   }
}
```

AndroidManifest.xml:

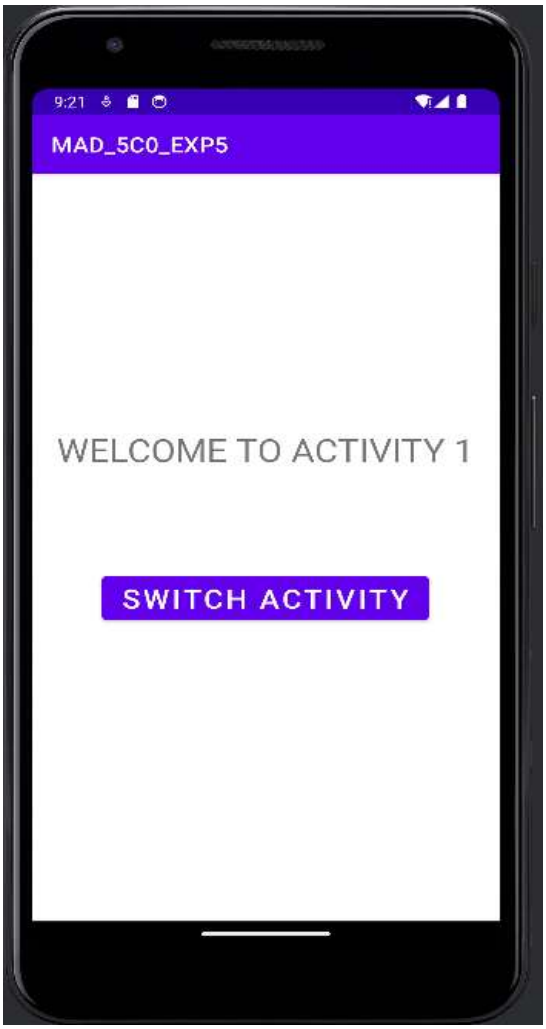
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
      android:dataExtractionRules="@xml/data_extraction_rules"
      android:fullBackupContent="@xml/backup_rules"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
      android:theme="@style/Theme.MAD_5C0_EXP5"
tools:targetApi="31">
<activity
android:name=".created"
android:exported="false" />
<activity
android:name=".main"
```

```
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:



ON CLICK OF **BUTTON**



SWITCHED ACTIVITY

EXPERIMENT-6

AIM:

Build mobile application to demonstrate Dynamic Fragments.

DESCRIPTION:

Dynamic Fragment is a type of fragment that is defined in an XML layout file and called using Fragment Manager class. The Fragment Manager class is responsible for managing fragments. It is a part of the Activity and its lifecycle depends on the lifecycle of its container activity. Dynamic Fragments are more responsive and flexible than Static Fragments.

Properties of Dynamic Fragment:

- Defined in Java class by extending Fragment Manager class.
- Having a fixed position in the Activity's layout but its content can be changed.
- Can be added, removed, or replaced at runtime.
- Created when the Activity is created and destroyed when the activity is destroyed.
- Step by Step Implementation
    1. Create a New Project in Android Studio
        a. To create a new project in Android Studio please refer to How to Create/Start a New Project in Android Studio. Note that select Java as the programming language.
    2. Working with the activity_main.xml file
        a. Navigate to the app > res > layout > activity_main.xml and add the below code to that file. Below is the code for the activity_main.xml file. Comments are added inside the code to understand the code in more detail.
    3. Working with Activity file (e.g. MainActivity.java)
        a. Here we call fragments using Fragment Manager class in Frame Layout.
    4. Working with Fragment layout (e.g. fragment_messages.xml)
    5. Working with Fragment layout (e.g. fragment_status.xml)
    6. Working with Fragment (e.g. MessagesFragment.java)
    7. Working with Fragment (e.g. StatusFragment.java)

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="http://schemas.android.com
/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
```

```
tools:context=".MainActivity">
<LinearLayout
      android:id="@+id/ll_bottom"
android:layout_width="302dp"
android:layout_height="296dp"
android:orientation="vertical"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.475"
app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/ll_top"
app:layout_constraintVertical_bias="0.394"></LinearLayout>
<LinearLayout
      android:id="@+id/ll_top"
android:layout_width="293dp"
android:layout_height="312dp"
android:layout_marginTop="96dp"
android:orientation="vertical"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

```
package com.example.fragments_5c0;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentResultListener;
import androidx.fragment.app.FragmentTransaction;
import android.os.Bundle;
import android.widget.ListView;
public class MainActivity extends AppCompatActivity {
   FragmentManager fmgr;
   FragmentTransaction ft;
   MilitaryFragment mf;
   ListView lv;
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
fmgr=getSupportFragmentManager();
mf=new MilitaryFragment();
fmgr.setFragmentResultListener("requestkey", this, new FragmentResultListener() {
        @Override
public void onFragmentResult(@NonNull String requestKey, @NonNull Bundle result) {
```

```
ft= fmgr.beginTransaction();
int option=result.getInt("SELECTED ITEM INDEX");
            switch (option) {
               case 0:
ft.replace(R.id.ll_bottom, new ArmyFragment());
                  break;
               case 1:
ft.replace(R.id.ll_bottom, new NavyFragment());
                  break;
               case 2:
ft.replace(R.id.ll_bottom, new AirForceFragment());
                  break;
default:ft.replace(R.id.ll_bottom,new ArmyFragment());
            }
ft.commit();
         }
      });
ft=fmgr.beginTransaction();
ft.add(R.id.ll_top,mf);ft.commit();
   }
}
```

military_fragment.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MilitaryFragment">
<ListView
      android:id="@+id/lv_military"
android:layout_width="match_parent"
android:layout_height="match_parent"
      android:entries="@array/military_list" />
</FrameLayout>
```

MilitaryFragent.java:

```
package com.example.fragments_5c0;
import android.os.Bundle;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
```

```
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.Toast;
public class MilitaryFragment extends Fragment implements
AdapterView.OnItemClickListener{
   ListView lv;
public MilitaryFragment() {
      // Required empty public constructor
   }
   @Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                        Bundle savedInstanceState) {
lv=container.findViewById(R.id.lv_military);
return inflater.inflate(R.layout.fragment_military, container, false);
   }
   @Override
public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
lv=(ListView)view.findViewById(R.id.lv_military);
      lv.setOnItemClickListener(this);
super.onViewCreated(view, savedInstanceState);
   }
   @Override
public void onItemClick(AdapterView<?> parent, View view, int i, long id) {
Toast.makeText(getContext(),lv.getItemAtPosition(i).toString(),Toast.LENGTH_SHORT).show
();
      Bundle bun=new Bundle();
bun.putInt("SELECTED ITEM INDEX",i);
getParentFragmentManager().setFragmentResult("requestkey",bun);

   }}
```

fragment_army.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".ArmyFragment">
<ImageView
      android:id="@+id/imageView4"
android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
      android:src="@drawable/img_3" />
</FrameLayout>
```

ArmyFragment.java:

```
package com.example.fragments_5c0;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
public class ArmyFragment extends Fragment {
   @Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                        Bundle savedInstanceState) {
      // Inflate the layout for this fragment
return inflater.inflate(R.layout.fragment_army, container, false);
   }
}
```

fragment_navy.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".NavyFragment">
<ImageView
      android:id="@+id/imageView"
android:layout_width="match_parent"
android:layout_height="match_parent"
      android:src="@drawable/img_1" />
</FrameLayout>
```

NavyFragment.java

```
package com.example.fragments_5c0;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
public class NavyFragment extends Fragment {
public NavyFragment()
```

```
    {
    }
    @Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
        // Inflate the layout for this fragment
return inflater.inflate(R.layout.fragment_navy, container, false);
    }
}
```

fragment_airforce.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".AirForceFragment">
<ImageView
        android:id="@+id/imageView2"
android:layout_width="match_parent"
android:layout_height="match_parent"
        android:src="@drawable/img_2" />
</FrameLayout>
```

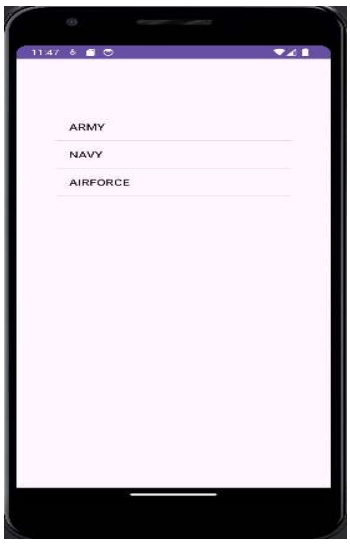
AirforceFragment.java:

```
package com.example.fragments_5c0;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
public class AirForceFragment extends Fragment {
public AirForceFragment() {
    }
    @Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
        // Inflate the layout for this fragment
return inflater.inflate(R.layout.fragment_air_force, container, false);
    }
}
```

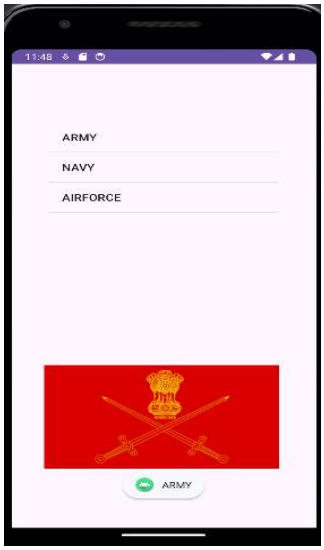
AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
      android:dataExtractionRules="@xml/data_extraction_rules"
      android:fullBackupContent="@xml/backup_rules"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
      android:theme="@style/Theme.FRAGMENTS_5C0"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

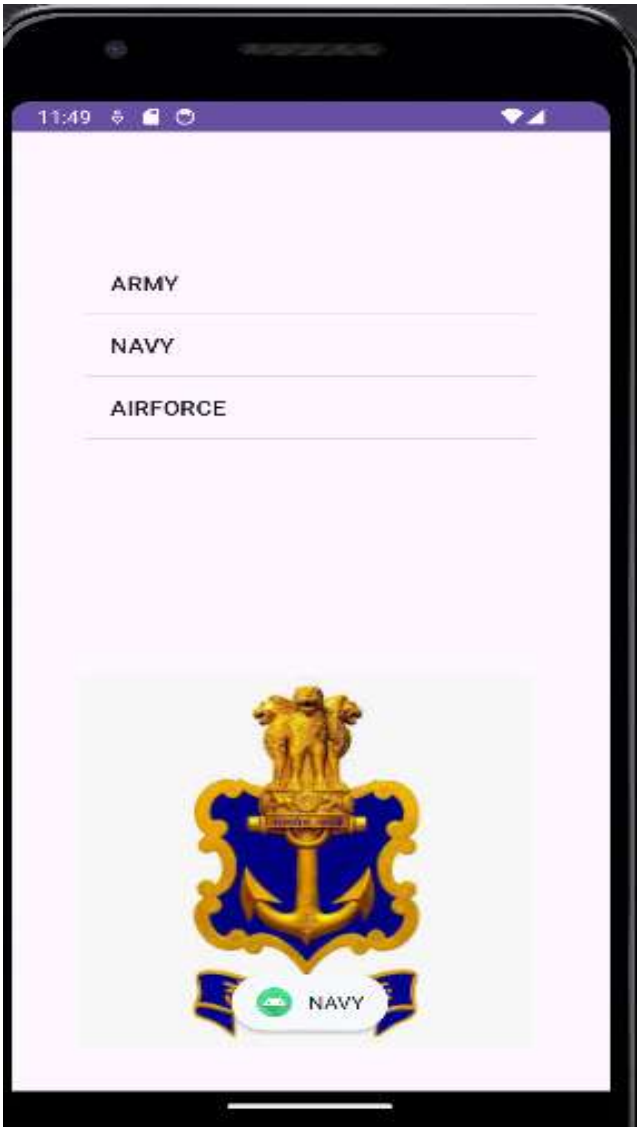
OUTPUT:



INITIAL DISPLAY



ON SELECT OF OPTION **ARMY**



ON SELECT OF OPTION **NAVY**

ON SELECT OF OPTION **AIRFORCE**

EXPERIMENT-7

AIM:

Build mobile application serverless database SQLite Database.

DESCRIPTION:

Android SQLite Database is a lightweight, embedded relational database management system that provides a convenient and efficient way to store and retrieve structured data in Android applications. SQLite database data are stored locally on the device, making them accessible even without an internet connection. Developers can use SQLite to persistently store data such as user preferences, app settings, or complex data structures. With features like transactions and indexes, SQLite in Android enables efficient data management and retrieval, making it a popular choice for local data storage in Android applications.

Importance of Using SQLite in Android development

SQLite is an essential tool for managing data storage and durability that is used extensively in Android development. Let's examine why SQLite is essential for Android development.

Embedded and lightweight:

SQLite is intended to be embedded, with a tiny footprint and little resource needs. It functions as an embedded database system, which means that no additional server is required for it to run on the device itself. Because of this feature, it is a great option for mobile systems like Android where resources are frequently scarce.

Local Data Storage:

Android SQLite enables programmers to keep data on the device locally. Applications that must run offline or with sporadic network access depend on this functionality to function. The user experience and productivity are improved by using SQLite, which enables programs to store and retrieve vital data without needing a continual network connection.

Structured Data Management:

SQLite offers a structured and well-organized method of managing data. Using SQL statements, developers may define tables, columns, and relationships, ensuring data consistency and integrity. The ability to structure data makes it simpler to deal with complicated datasets and carry out advanced data operations by enabling efficient data manipulation, querying, and retrieval.

SQL Compatibility:

The SQLite database in Android is built on the SQL (Structured Query Language) standard. The database is compatible with SQL, allowing developers to take advantage of their current SQL knowledge and expertise. Developers may easily construct sophisticated queries and carry out a wide range of actions thanks to SQL's robust and standardized syntax for interacting with databases.

Transaction Support:

SQLite database in android offers transaction management, enabling programmers to combine several database operations into a single, atomic operation. All of the processes contained inside a transaction must be completed for it to be successful, or none of them must be applied. Particularly when handling concurrent operations or intricate data manipulations, this atomicity attribute aids in maintaining data integrity and consistency.

Indexing and query optimization:

By utilizing indexes and query optimization techniques, SQLite database in android offers effective data retrieval. Developers can speed up data retrieval processes by building indexes on frequently used columns. By choosing the most effective execution plan, the query optimizer in SQLite uses cost-based query planning and execution to guarantee optimal query performance.

Widely Used and Battle-Tested:

One of the most used database engines in the world, SQLite is renowned for its dependability, stability, and performance. It is a dependable option for data storage in Android applications because it has undergone considerable testing and optimization over the years. Because of its popularity, there is a sizable development community and copious documentation, which makes it simpler to locate help and resources.

Security and Privacy:

To protect the security and privacy of important data, SQLite database in android are always maintained in an encrypted manner. An additional degree of security against unauthorized access is provided by the fact that an SQLite database's data can only be viewed by the program that generated it.

Explanation of SQLiteOpenHelper class

Two crucial methods, onCreate() and onUpgrade(), must be overridden when building a subclass of SQLiteOpenHelper. These methods, which the SQLiteOpenHelper class calls at certain times in its lifecycle, let you provide unique behavior for creating and updating databases.

onCreate():

When a database is first built, the onCreate() function is invoked. Its duties include running SQL queries to create tables, columns, constraints, and indexes as well as other operations that construct the database structure. The relevant SQL queries for building the database structure are executed inside of this function using the execSQL() method of the SQLiteDatabase class.

onUpgrade():

The onUpgrade() function is invoked whenever a version update necessitates an upgrade of the database. It enables you to specify the database schema modifications that will be necessary to support application updates. By using this strategy, you may often change tables, add or remove columns, or migrate data from one schema to another. To avoid data loss or corruption, it is essential to properly manage data migration during database updates.

Understanding the SQLiteOpenHelper lifecycle

Initialization:

The onCreate() function of the application or as necessary creates an instance of the SQLiteOpenHelper subclass. The database name and version are entered into the constructor of the SQLiteOpenHelper class at this point.

First-time Database Creation:

The onCreate() function is called if the database does not already exist or if the version supplied is greater than the version of the installed database. Here, you design the database schema and run the required SQL operations to build the basic framework.

Database Opening:

The onOpen() method is called before returning the SQLiteDatabase instance when you use the getWritableDatabase() or getReadableDatabase() methods of the SQLiteOpenHelper class to request a writable or readable instance of the database. This enables you to carry out any startup or setup procedures particular to each database session.

Database Upgrade:

The onUpgrade() method is triggered if the SQLiteOpenHelper subclass specifies a database version that is greater than the one that is currently in use. This approach takes care of updating the database schema to take into account the adjustments the new version demands. Tables may need to be changed, columns may need to be changed, or data may need to be moved from the old schema to the new one.

Normal Database Use:

Using the provided SQLiteDatabase instance, you may carry out standard database activities including data insertion, retrieval, updating, and deletion after the database has been built or updated. Using the methods offered by the SQLiteDatabase class, these activities are often carried out within the application's code.

SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:onClick="deleteStudent"
tools:context=".MainActivity">
<TextView
    android:id="@+id/tv_title"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
    android:text="SQLite Database"
    android:textColor="@color/purple_200"
android:textSize="24sp"
android:textStyle="bold"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"></TextView>
<EditText
    android:id="@+id/et_roll"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/tv_title"
    android:text="Roll"
    android:textColor="@color/teal_700"
android:textSize="24sp"
android:textStyle="bold"></EditText>
<EditText
    android:id="@+id/et_name"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/et_roll"
```

```
    android:text="Name"
    android:textColor="@color/teal_700"
android:textSize="24sp"
android:textStyle="bold"></EditText>
<EditText
    android:id="@+id/et_avg"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/et_name"
    android:text="Average"
    android:textColor="@color/teal_700"
android:textSize="24sp"
android:textStyle="bold"></EditText>
<EditText
    android:id="@+id/et_Grade"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/et_avg"
    android:text="Grade"
    android:textColor="@color/teal_700"
android:textSize="24sp"
android:textStyle="bold"></EditText>
<Button
    android:id="@+id/btn_insert"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/et_Grade"
android:onClick="insertStudent"
    android:text="Insert"
android:textSize="24sp"></Button>
<Button
    android:id="@+id/btn_update"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/btn_insert"
android:onClick="updateStudent"
    android:text="Update "
android:textSize="24sp"></Button>
<Button
    android:id="@+id/btn_delete"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/btn_update"
android:onClick="deleteStudent"
    android:text="Delete"
android:textSize="24sp"></Button>
```

```
<Button
    android:id="@+id/btn_view"
android:layout_width="match_parent"
android:layout_height="wrap_content"
    android:layout_below="@id/btn_delete"
android:onClick="viewStudent"
    android:text="View"
android:textSize="24sp"></Button>
</RelativeLayout>
```

MainActivity.java:

```
package com.example.21501a05C0_exp7;
import androidx.appcompat.app.AppCompatActivity;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
StudentDBHelper studentDBHelper;
EditText et_roll,et_name,et_avg,et_Grade;
   @Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
       studentDBHelper=new StudentDBHelper(this);
       et_roll=this.findViewById(R.id.et_roll);
       et_name=this.findViewById(R.id.et_name);
       et_avg=this.findViewById(R.id.et_avg);
       et_Grade=this.findViewById(R.id.et_Grade);
   }
public void insertStudent(View v)
   {
       String roll=et_roll.getText().toString();
       String name=et_name.getText().toString();
       String avg=et_avg.getText().toString();
       String grade=et_Grade.getText().toString();
boolean res=studentDBHelper.insertstudent(roll,name,avg,grade);
       if(res)
       {
Toast.makeText(this, "Insert Success", Toast.LENGTH_LONG).show();
       }
       else {
Toast.makeText(this, "Insert Failed", Toast.LENGTH_LONG).show();
       }
```

```
    }
public void updateStudent(View v)
    {
        String roll=et_roll.getText().toString();
        String name=et_name.getText().toString();
        String avg=et_avg.getText().toString();
        String grade=et_Grade.getText().toString();
boolean res=studentDBHelper.updatestudent(roll,name,avg,grade);
        if(res)
        {
Toast.makeText(this, "Update Success", Toast.LENGTH_LONG).show();
        }
        else {
Toast.makeText(this, "Update Failed", Toast.LENGTH_LONG).show();
        }
    }
public void deleteStudent(View v)
    {
        String roll=et_roll.getText().toString();
        boolean res=studentDBHelper.deletestudent(roll);
        if(res)
        {
Toast.makeText(this, "Delete Success", Toast.LENGTH_LONG).show();
        }
        else {
Toast.makeText(this, "Delete Failed", Toast.LENGTH_LONG).show();
        }
    }
public void viewStudent(View v)
    {
        String roll=et_roll.getText().toString();
        studentDBHelper.viewStudent(roll);
        Cursor student=studentDBHelper.viewStudent(roll);
if(student.moveToFirst())
        {
            et_roll.setText(roll);
            et_name.setText(student.getString(1));
            et_avg.setText(student.getString(2));
            et_Grade.setText(student.getString(3));
        }
        else
        {
Toast.makeText(this, "No Such Student", Toast.LENGTH_LONG).show();
        }
    }
}
```

StudentdbpHelper.java:

```
package com.example.21501a05C0_exp7;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import androidx.annotation.Nullable;
public class StudentDBHelper extends SQLiteOpenHelper {
public StudentDBHelper(Context context) {
super(context,"student.db", null, 1);
   }
   @Override
public void onCreate(SQLiteDatabase sqLiteDatabase) {
sqLiteDatabase.execSQL("create table StudentGrade(roll TEXT primary key,name TEXT ,avg
TEXT,grade TEXT)");
   }
   @Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {
      sqLiteDatabase.execSQL("drop table if exists StudentGrade");
   }
public boolean insertstudent(String r,String n,String a,String g)
   {
      SQLiteDatabase sqLiteDatabase=this.getWritableDatabase();
      ContentValues student=new ContentValues();
student.put("roll",r);
student.put("name",n);
student.put("avg",a);
student.put("grade",g);
long res=sqLiteDatabase.insert("StudentGrade",null,student);
      if(res==-1)
      {return false;}
      else return true;
   }
public boolean updatestudent(String r,String n,String a,String g)
   {
      SQLiteDatabase sqLiteDatabase=this.getWritableDatabase();
      ContentValues student=new ContentValues();
student.put("roll",r);
student.put("name",n);
student.put("avg",a);
student.put("grade",g);
String[] where=new String[]{r};
```

```
int res=sqLiteDatabase.update("StudentGrade",student,"roll=?",where);
      if(res>0)
      {return false;}
      else return true;
   }
public boolean deletestudent(String r)
   {
      SQLiteDatabase sqLiteDatabase=this.getWritableDatabase();
      ContentValues student=new ContentValues();
student.put("roll",r);
String[] where=new String[]{r};
int res=sqLiteDatabase.delete("StudentGrade","roll=?",where);
      if(res>0)
      {return true;}
      else return false;
   }
public Cursor viewStudent(String r){
      SQLiteDatabase sqLiteDatabase=this.getWritableDatabase();
String[] where=new String[]{r};
return sqLiteDatabase.rawQuery("select * from StudentGrade where roll=?",where);
   }
}
```

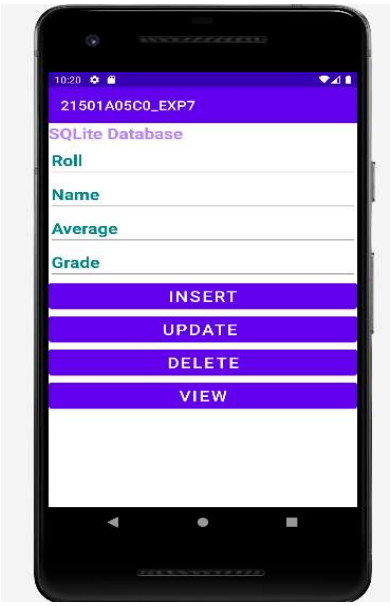
AndroidManifest.java:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
     android:dataExtractionRules="@xml/data_extraction_rules"
     android:fullBackupContent="@xml/backup_rules"
     android:icon="@mipmap/ic_launcher"
     android:label="@string/app_name"
     android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
     android:theme="@style/Theme._21501A05C0_EXP7"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
```

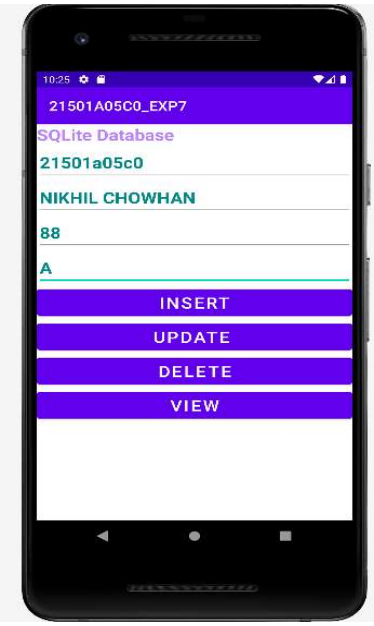
```
</application>
</manifest>
```

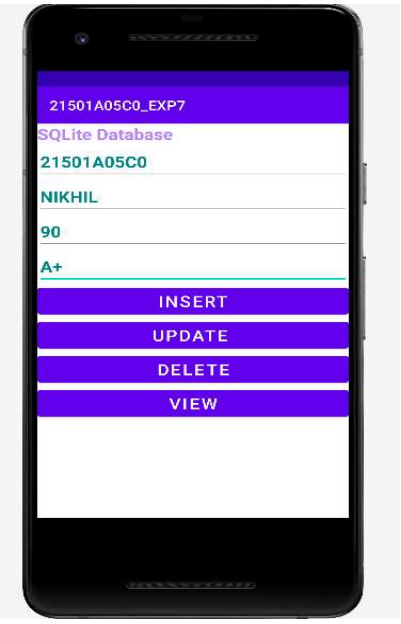
OUTPUT:

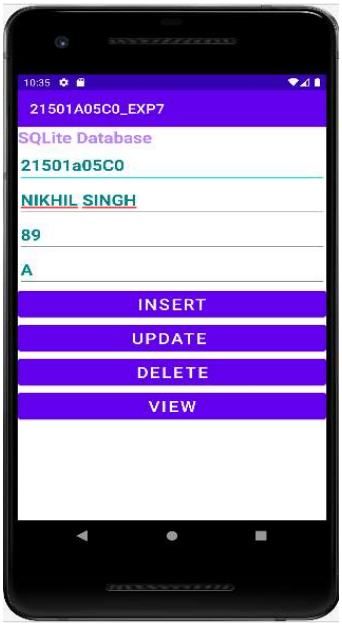
//INITIAL DISPLAY



//INSERT OPERATION

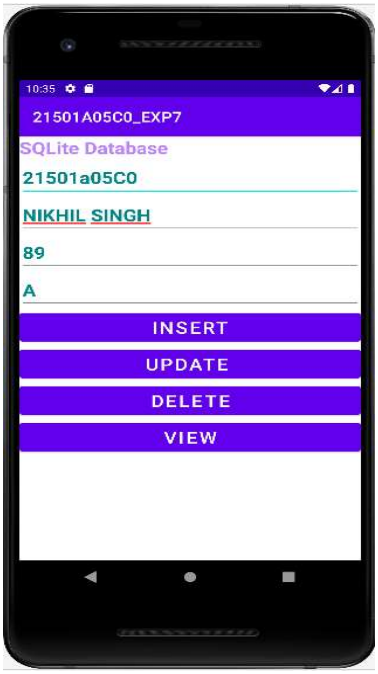






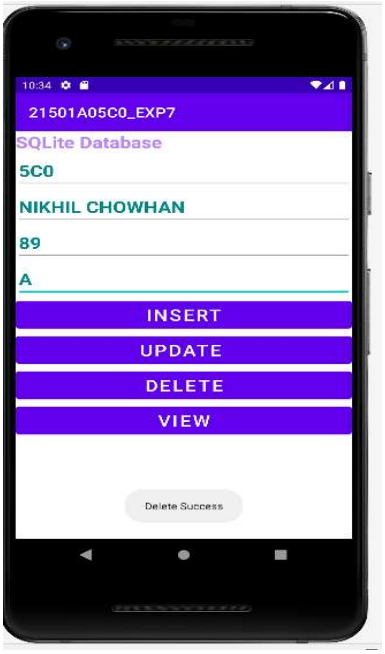
roll	name	avg	grade
21501A05C0	NIKHIL	90	A+
21501a05C0	NIKHIL SIN...	89	A
21501a05c0	NIKHIL CH...	88	A

/UPDATE OPERATION



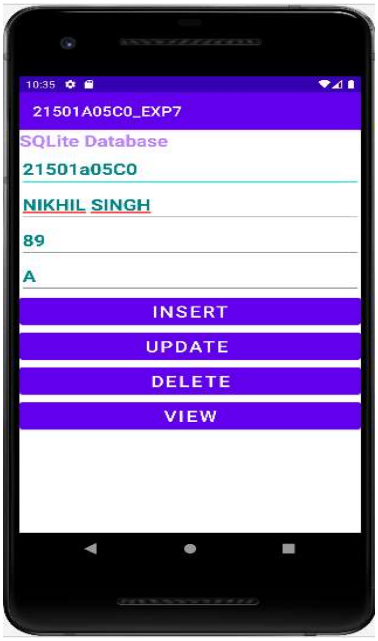
roll	name	avg	grade
21501a05C0	NIKHIL SIN...	89	A
21501a05c0	NIKHIL CH...	88	A
5C0	NIKHIL CH...	89	A

//DELETE OPERATION:



roll	name	avg	grade
21501a05C0	NIKHIL SIN...	89	A
21501a05c0	NIKHIL CH...	88	A

//VIEW



EXPERIMENT-7[B]

AIM:

Build mobile application serverless database Firebase (cloud-hosted database)

DESCRIPTION:

Firebase is a mobile platform that helps you quickly develop high-quality apps, grow your user base, and earn more money. Firebase consists of complementary features that you can mix and match to fit your needs, with Google Analytics for Firebase at the core. You can explore and integrate Firebase services in your app directly from Android Studio using the **Assistant** window shown in figure 1.

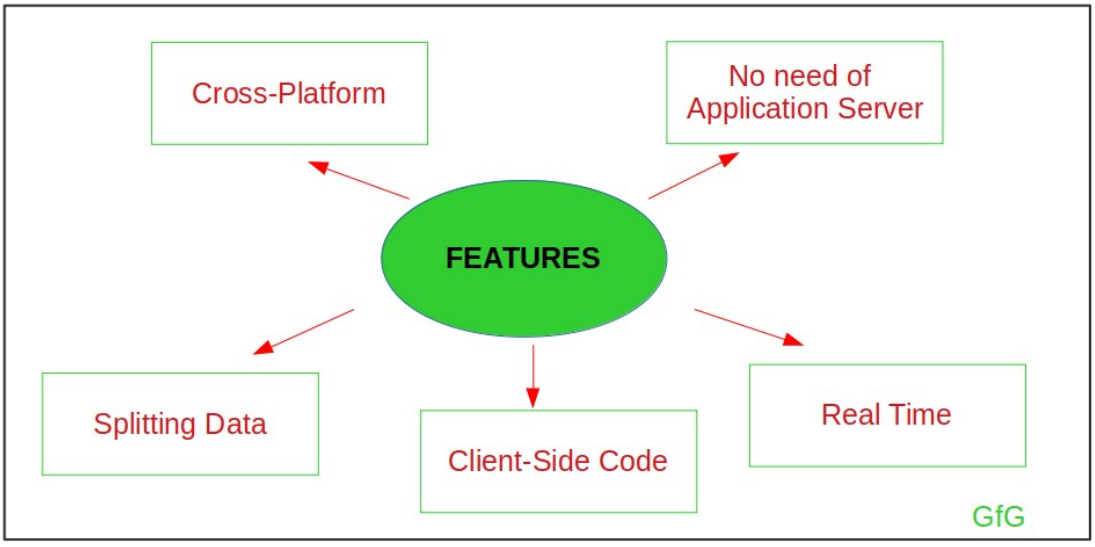
First, make sure you have added Google's Maven repository to your project configuration.

You can open and use the **Assistant** window in Android Studio by following these steps:

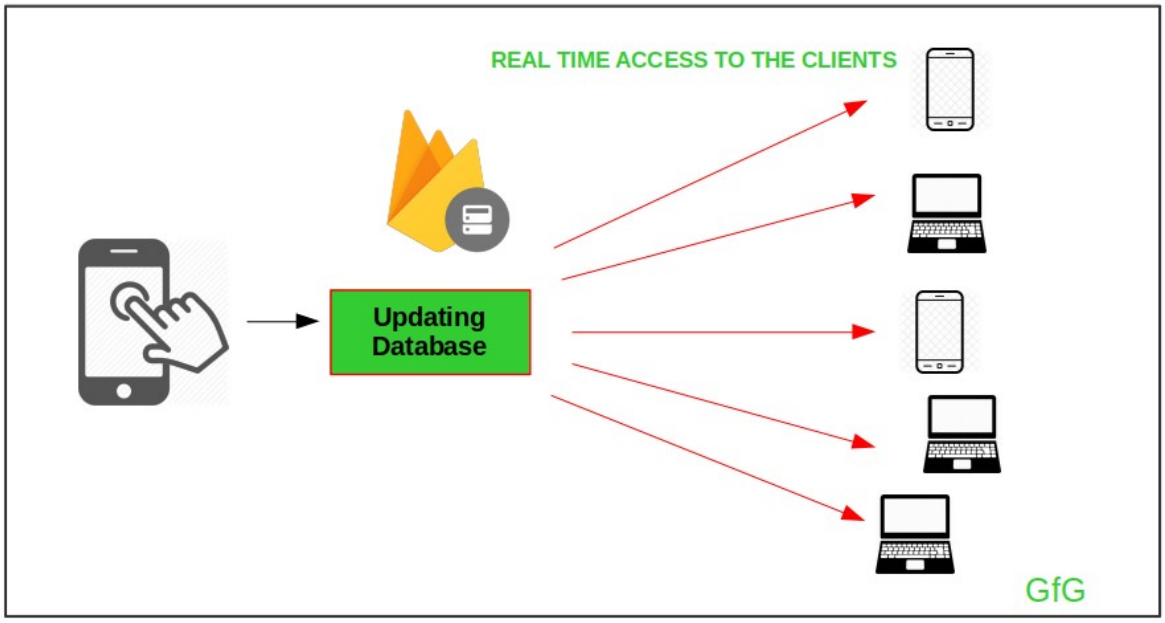
1. Select **Tools > Firebase** to open the **Assistant** window.
2. Click to expand one of the listed features.
3. Click **Get Started with Firebase Analytics** to open a tutorial that connects you to Firebase and adds the necessary code to your app.

Firebase Realtime Database is a Cloud hosted database, i.e. it runs on a cloud and access to the user is provided as a service. It stores data in JSON (Javascript Object Notation) format, a format to store or transport data. All the users connected to it can get access to the data at Real Time.

Features of Firebase Realtime Database?

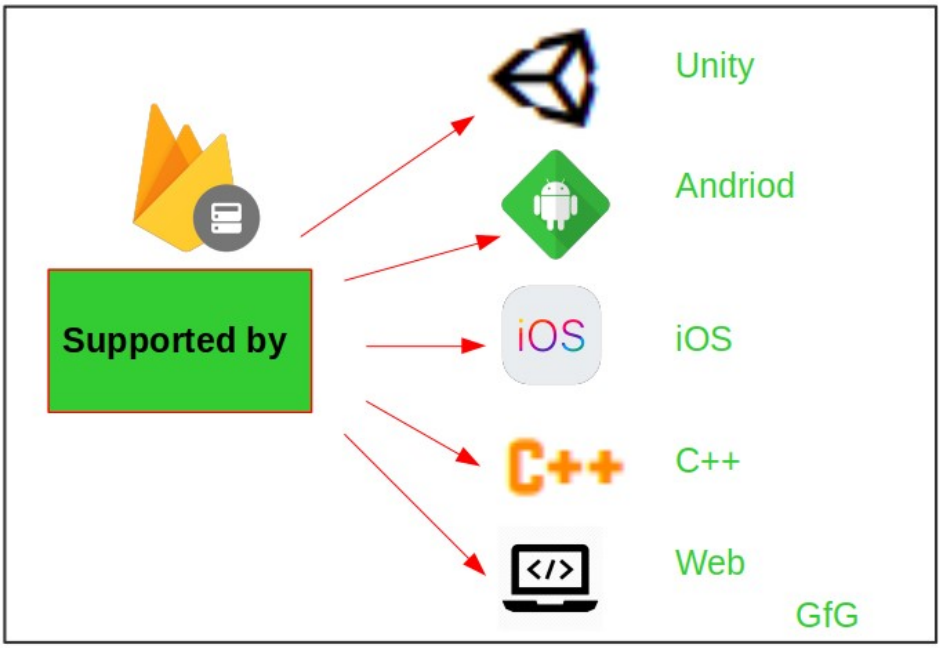


1.Real Time: Due to the Data synchronization used in Real Time, every update is received by the devices/clients in no time.



2.No need of Application Server: As the database can be accessed directly from the mobile device or browser there is no need for an Application Server.

3.Support by various languages and platforms:



4.Splitting Data: The client can split the data across multiple database instances for the same project.

5.Client-Side Code: The dynamic applications with the secured data can be accessed directly from the client-side code.

6.Cross-Platform: It can be used for building a back-end for various platforms like Android, iOS, Web, iOS as well as JavaScript SDK.

Structuring the Realtime Database:

Firebase Realtime Database stores data as one large JSON tree. It stores simple data easily, but the unstructured hierarchical data is difficult to organize. Unlike SQL there are no tables here.

What is JSON Tree Model?

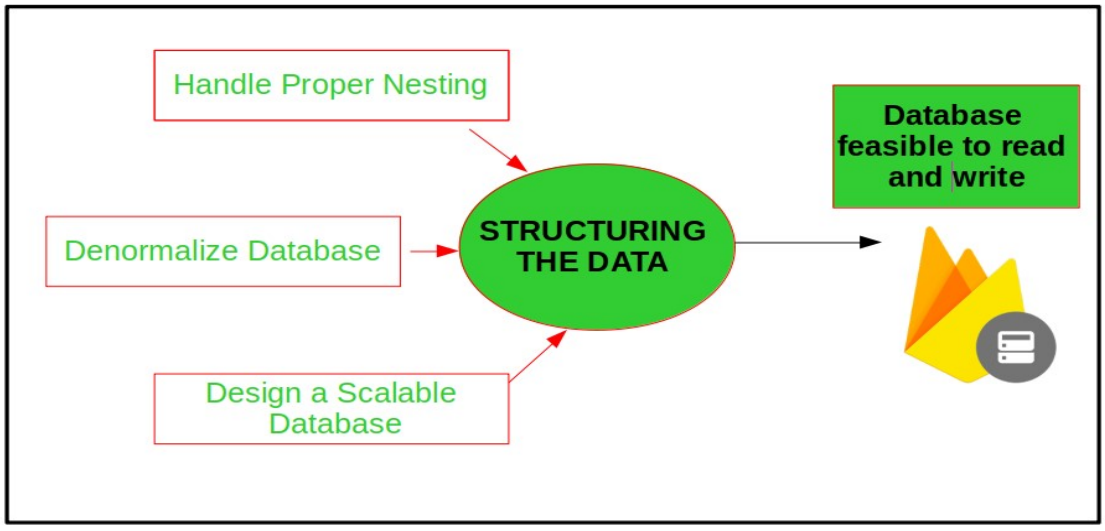
- JSON Tree Model is inspired from JSON(JavaScript Object Notation) used to represent the JSON document which generally consists of key-value pair in memory .

Why to structure the data (What are the advantages of doing it)?

- If data is stored in well formatted structure then it is easy to save as well as retrieve it easily.
- Querying becomes easy on the structured data.
- It becomes feasible to refer the data in structured format.

Key points to remember while structuring the data:

Before writing and reading the data into the database, the main aim of the developer should be constructing the structure of the database.



SOURCE CODE:

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">
<ListView
```

```
      android:id="@+id/lv_cities"
android:layout_width="409dp"
android:layout_height="354dp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintVertical_bias="0.289" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

```
package com.example.exp7_firebase_5c0;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.GenericTypeIndicator;
import com.google.firebase.database.ValueEventListener;
import java.util.ArrayList;
import java.util.Map;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.GenericTypeIndicator;
import com.google.firebase.database.ValueEventListener;
import java.util.ArrayList;
import java.util.Map;
public class MainActivity extends AppCompatActivity {
   ListView lv_cities;
   ArrayAdapter<String> aadap;
   ArrayList<String> arr_list;
protected void onCreate(Bundle savedInstanceState) {
```

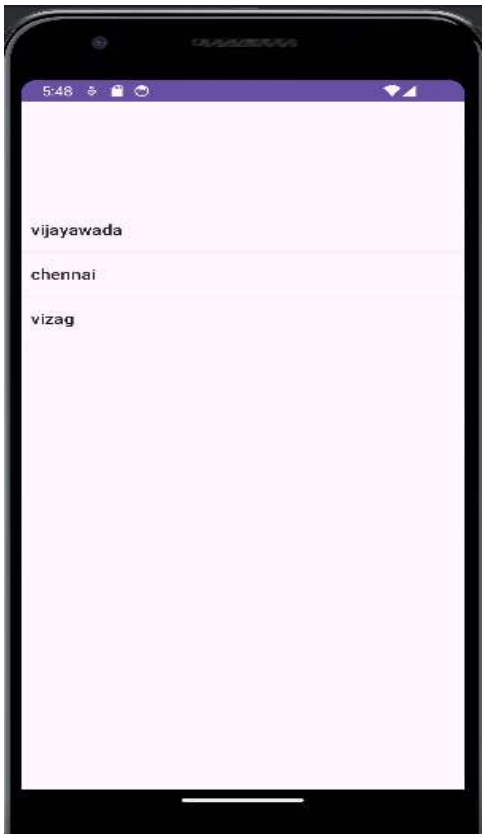
```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
      lv_cities = findViewById(R.id.lv_cities);
      arr_list = new ArrayList<String>(); // Initialize ArrayList before creating ArrayAdapter
aadap = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1, arr_list);
      DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReference("cities");
databaseReference.addValueEventListener(new ValueEventListener() {
         @Override
public void onDataChange(@NonNull DataSnapshot snapshot) {
if (snapshot.exists()) {
Toast.makeText(getApplicationContext(), "hi", Toast.LENGTH_LONG).show();
               GenericTypeIndicator<Map<String, String>> genericTypeIndicator = new
GenericTypeIndicator<Map<String, String>>() {};
               Map<String, String> map = snapshot.getValue(genericTypeIndicator);
if (map != null) {
for (Map.Entry<String, String> map1 : map.entrySet()) {
                     arr_list.add(map1.getValue());
                  }
aadap.notifyDataSetChanged(); // Move notifyDataSetChanged() here
               }
            }
         }
         @Override
public void onCancelled(@NonNull DatabaseError error) {
            // Handle onCancelled
         }
      });
      lv_cities.setAdapter(aadap); // Set adapter after adding data to the ArrayList
   }}
```

AndroidManifest.java:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools">
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"></uses-
permission>
<application
android:allowBackup="true"
      android:dataExtractionRules="@xml/data_extraction_rules"
      android:fullBackupContent="@xml/backup_rules"
      android:icon="@mipmap/ic_launcher"
      android:label="@string/app_name"
      android:roundIcon="@mipmap/ic_launcher_round"
```

```
android:supportsRtl="true"
      android:theme="@style/Theme.Exp7_firebase_5c0"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:



EXPERIMENT – 8

AIM:

Build mobile application based on the Google Maps

DESCRIPTION:

Google Maps is a web service that provides detailed information about geographical regions and sites worldwide. In addition to conventional road maps, Google Maps offers aerial and satellite views of many locations. In some cities, Google Maps offers street views comprising photographs taken from vehicles.

MainActivity.java:

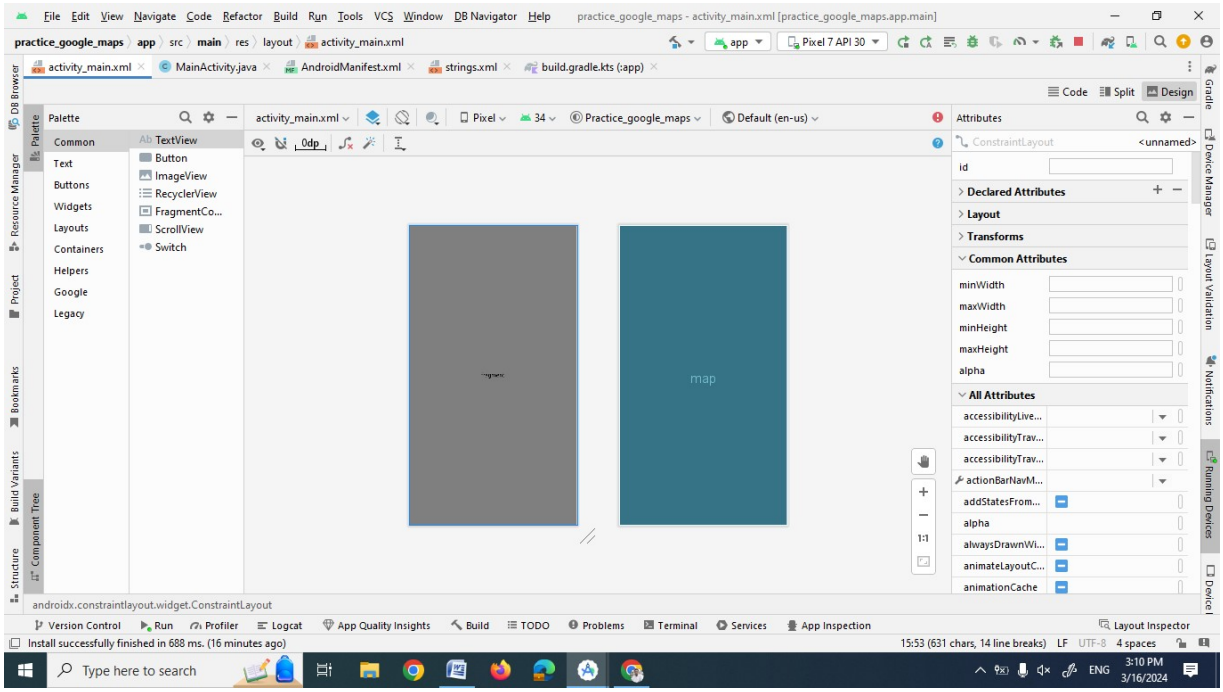
```
package com.example.practice_google_maps;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class MainActivity extends AppCompatActivity implements OnMapReadyCallback {
GoogleMap gMap;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      SupportMapFragment supportMapFragment=(SupportMapFragment)
            getSupportFragmentManager().findFragmentById(R.id.map);
      supportMapFragment.getMapAsync(this);
   }
@Override
public void onMapReady(@NonNull GoogleMap googleMap)
   {
gMap=googleMap;
      LatLng vja=new LatLng(16.48816,80.69413);
      MarkerOptions mo=new MarkerOptions();
      mo.position(vja);
      mo.title("PVPSIT");
gMap.addMarker(mo);
gMap.moveCamera(CameraUpdateFactory.newLatLng(vja));
gMap.getUiSettings().setZoomControlsEnabled(true);
```

```
gMap.getUiSettings().setCompassEnabled(true);}}
```

activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">
<fragment
android:id="@+id/map"
android:name="com.google.android.gms.maps.SupportMapFragment"
android:layout_width="match_parent"
android:layout_height="match_parent"/>
</androidx.constraintlayout.widget.ConstraintLayout>
```



AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools">
<application
android:allowBackup="true"
```

```
android:dataExtractionRules="@xml/data_extraction_rules"
android:fullBackupContent="@xml/backup_rules"
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true"
android:theme="@style/Theme.Practice_google_maps"
tools:targetApi="31">
<activity
android:name=".MainActivity"
android:exported="true">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<meta-data
android:name="com.google.android.geo.API_KEY"
android:value="@string/mic_maps_api_key" />
</application>
</manifest>
```

Strings.xml

```
<resources>
<string name="app_name">practice_google_maps</string>
<string name="mic_maps_api_key">AIzaSyBtNldqVCKeU9eK-Q_eyZE0iHP-
C4RLY5o</string>
</resources>
```

Build.gradle.kts(:app):

```
plugins {
id("com.android.application")
}
android {
namespace = "com.example.practice_google_maps"
compileSdk = 34
defaultConfig {
applicationId = "com.example.practice_google_maps"
minSdk = 24
targetSdk = 34
versionCode = 1
versionName = "1.0"

testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
}
```

```
buildTypes {
release {
isMinifyEnabled = false
proguardFiles(getDefaultProguardFile("proguard-android-optimize.txt"), "proguard-
rules.pro")
}}
compileOptions {
sourceCompatibility = JavaVersion.VERSION_1_8
targetCompatibility = JavaVersion.VERSION_1_8  }}
dependencies {
implementation("androidx.appcompat:appcompat:1.6.1")
implementation("com.google.android.material:material:1.11.0")
implementation("androidx.constraintlayout:constraintlayout:2.1.4")
testImplementation("junit:junit:4.13.2")
androidTestImplementation("androidx.test.ext:junit:1.1.5")
androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
implementation("com.google.android.gms:play-services-maps:18.2.0")
}
```

OUTPUT: